New efficient substepping methods for exponential timestepping

Applied Mathematics and Computation 307 (2017) 342–365

Contents lists available at ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

New efficient substepping methods for exponential timestepping

G.J. Lord, D. Stone*

Department of Mathematics & Maxwell Institute, Heriot-Watt University, Edinburgh, United Kingdom

ARTICLE INFO

Keywords:
Exponential integrators
Krylov subspace methods
Advection-diffusion-reaction equations

ABSTRACT

Exponential integrators are time stepping schemes which exactly solve the linear part of a semilinear ODE system. This class of schemes requires the approximation of a matrix exponential in every step, and one successful modern method is the Krylov subspace projection method. We investigate the effect of breaking down a single timestep into arbitrary multiple substeps, recycling the Krylov subspace to minimise costs. For these recycling based schemes we analyse the local error, investigate them numerically and show they can be applied to a large system with 10^6 unknowns. We also propose a new second order integrator that is found using the extra information from the substeps to form a corrector to increase the overall order of the scheme. This scheme is seen to compare favorably with other order two integrators.

1. Introduction

We consider the numerical integration of a large system of semilinear ODEs of the form

$$\frac{du}{dt} = Lu + F(t, u(t)) \qquad u(0) = u_0, \quad t \in [0, \infty) \tag{1}$$

with $u, F(t, u(t)) \in \mathbb{R}^N$ and $L \in \mathbb{R}^{N\times N}$ a matrix. Eq. (1) arises, for example, from the spatial discretisation of reaction-diffusion-advection equations. An increasingly popular method for approximating the solution of semlinar ODE systems such as (1) are exponential integrators. These are a class of schemes which approximate (1) by exactly solving the linear part and are characterised by requiring the evaluation or approximation of a matrix exponential function of L at each timestep. A major class of exponential integrators are the multistep exponential time differencing (ETD) schemes, first developed in [1], other classes include the exponential Euler midpoint method [2] and exponential Rosenbrock type methods [3,4]. For an overview of exponential integrators see [5,6] and other useful references can be found in [7].

Exponential integrators potentially have several significant advantages over traditional implicit integrators. They often have favourable stability properties (see for example the analysis in Section 3 in [1]), which allows for larger timesteps; they work well without preconditioning, and are simple to implement when a method for approximating the necessary matrix exponential functions is in place (see below). Investigations have shown exponential integrators to be competitive with, or to outperform in some cases, more traditional methods; see for example [8–11].

* Corresponding author.
E-mail addresses: G.J.Lord@hw.ac.uk (G.J. Lord), D.Stone@hw.ac.uk, daniel.stone.work@gmail.com (D. Stone).

http://dx.doi.org/10.1016/j.amc.2017.02.052
0096-3003/

Approximating the matrix exponential and functions of it (like $\varphi-$functions in (3) below) is a notoriously difficult problem [12]. A classical technique is Padé approximation, which is only efficient for small matrices. Modern methods range from Taylor series methods making sophisticated use of scaling and squaring for efficiency, [13], to approximation with Faber or Chebyschev polynomials [5,14] Section 4.1, interpolation on Leja points [15–19], to Krylov subspace projection techniques [20–24] which is what we consider here.

Our schemes are based on the standard exponential integrator ETD1, which can be written as

$$u_{n+1}^{etd} = u_n^{etd} + \Delta t \varphi_1(\Delta t L)\left(Lu_n^{etd} + F_n^{etd}\right), \tag{2}$$

where $\varphi_1(\Delta tL)$ is defined shortly; $u_n^{etd} \approx u(t_n)$ at discrete times $t_n = n\Delta t$ for fixed $\Delta t > 0$, $F_n^{etd} \equiv F(t_n, u_n^{etd})$ and $n \in \mathbb{N}$. ETD1 is globally first order, and is derived from (1) by variation of constants and approximating $F(t, u(t))$ by the constant F_n^{etd} over one timestep. See for example [5–7,25] for more detail. It is useful to introduce the additional notation

$$g(t) \equiv Lu(t) + F(t, u(t)) \quad \text{and} \quad g_n^{etd} \equiv Lu_n^{etd} + F(t_n, u_n^{etd}).$$

The function φ_1 is part of a family of matrix exponential functions defined by $\varphi_0(z) = e^z$, $\varphi_1(z) = z^{-1}(e^z - I)$, and in general

$$\varphi_{k+1}(z) = z^{-1}\left(\varphi_k - \frac{I}{k!}\right), \tag{3}$$

where I is the identity matrix. These $\varphi-$functions appear in all exponential integrator schemes; see [23]. In particular we use φ_1, and for brevity we introduce the following notation

$$p_\tau \equiv \tau\varphi_1(\tau L). \tag{4}$$

We can then re-write (2) as

$$u_{n+1}^{etd} = u_n^{etd} + p_{\Delta t} g_n^{etd} \tag{5}$$

We consider the Krylov projection method for approximating terms like $p_{\Delta t} g_n^{etd}$ in (2). In the Krylov method, this term is approximated on a Krylov subspace defined by the vector g_n and the matrix L. Typically the subspace is recomputed, in the form of a matrix of basis vectors V_m, every time the solution vector, u_n in (2), is updated (and thus also g_n). This is done using a call to the Arnoldi algorithm (see for example the algorithm at the start of Section 2.1 in [20], or Algorithm 1 in [23]), and is often the most expensive part of each step. It is possible to 'recycle' this matrix at least once, as demonstrated in [26] for the exponential Euler method (EEM) (see [5] and (B.1)). In this paper we investigate this possibility further and use it to construct new methods based on ETD1 and in Appendix B we show how to construct the general recycling method for EEM.

We examine the effect of splitting the single step of (2) of length Δt in to S substeps of length $\delta t = \frac{\Delta t}{S}$, through which the Krylov subspace and its associated matrices are recycled. By deriving expressions for the local error, we show that the scheme remains locally second order for any number S of substeps, and that the leading term of the local error decreases. This gives a method based on recycling the Krylov subspace for S substeps. We then obtain a second method using the extra information from the substeps to form a corrector to increase the overall order of the scheme.

The paper is arranged as follows. In Section 2, we describe the Krylov subspace projection method for approximating the action of $\varphi-$functions on vectors. In Section 3, we describe the concept of recycling the Krylov subspace across substeps in order to increase the accuracy of the ETD1 based scheme, and show that the leading term of the local error of the scheme decreases as the number of substeps uses increases. We then prove a lemma to express the local error expression at arbitrary order. With this information about the local error expansion, and the extra information from the substeps taken, it is possible to construct correctors for the scheme the increase the accuracy and local order of the scheme. We demonstrate one simple such corrector in Section 4. Numerical examples demonstrating the effectiveness of this scheme are presented in Section 5.

2. The Krylov subspace projection method and ETD1

We describe the Krylov subspace projection method for approximating $\varphi_1(\Delta tL)$ in (2). We motivate this by showing how the leading powers of ΔtL in L are captured by the subspace. The series definition of $\varphi_1(\Delta tL)$ is,

$$\varphi_1(\Delta tL) \equiv \sum_{k=0}^{\infty} \frac{(\Delta tL)^k}{(k+1)!}. \tag{6}$$

The challenge in applying the scheme (2) is to efficiently compute, or approximate, the action of φ_1 on the vector g_n^{etd}. The sum in (6) is useful in motivating a polynomial Krylov subspace approximation. The m-dimensional Krylov subspace for the matrix L and vector $g \in \mathbb{R}^N$ is defined by:

$$\mathcal{K}_m(L, g) = \text{span}\{g, Lg, \ldots, L^{m-1}g\}. \tag{7}$$

Approximating the sum in (6) by the first m terms is equivalent to approximation in the subspace $\mathcal{K}_m(L, g_n^{etd})$ in (7). We now review some simple results about the general subspace $\mathcal{K}_m(L, g)$, with arbitrary vector g, before using the results with $g = g_n^{etd}$ to demonstrate how they are used in the evaluation of (2).

The Arnoldi algorithm (again see e.g. [20,23]) is used to produce an orthonormal basis $\{v_1, \ldots, v_n\}$ for the space $\mathcal{K}_m(L, g)$ such that

$$\text{span}\{v_1, v_2, \ldots, v_m\} = \text{span}\{g, Lg, \ldots, L^{m-1}g\}. \tag{8}$$

It produces two matrices $V_m \in \mathbb{R}^{N\times m}$, whose columns are the v_k, and an upper Hessenburg matrix $H_m \in \mathbb{R}^{m\times m}$. The matrices L, H_m and V_m are related by

$$LV_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T \tag{9}$$

where $h_{m+1,m}$ is an entry of H_m that the $m+1$th step of the Arnoldi algorithm would have produced, and similarly v_{m+1} is the $m+1$th orthogonal basis vector that would have been produced by that step. The e_m is the standard unit mth unit vector. Eq. (9) is (2) in [20]; see that reference and specifically Section 2 there for more detail.

By left multiplying (9) by V_m^T and using the fact that v_{m+1} is orthogonal with all the columns of V_m, we arrive at the relation,

$$H_m = V_m^T L V_m. \tag{10}$$

From (10) it follows that,

$$V_m H_m V_m^T = V_m V_m^T L V_m V_m^T. \tag{11}$$

For any $x \in \mathcal{K}_m(L, g)$,

$$V_m V_m^T x = x, \tag{12}$$

since $V_m V_m^T x$ represents the orthogonal projection into the space $\mathcal{K}_m(L, g)$. Therefore, since $L^k g \in \mathcal{K}_m(L, g)$, we also have that

$$V_m V_m^T L^k g = L^k g \qquad \text{for} \qquad 0 \le k \le m-1 \tag{13}$$

We now consider the relationship between $L^k g$ and $V_m H_m^k V_m^T g$.

Lemma 2.1. *Let $0 \le k \le m-1$. Then for H_m, V_m corresponding to the Krylov subspace $\mathcal{K}(L, g)$,*

$$V_m H_m^k V_m^T g = L^k g. \tag{14}$$

Proof. By induction. For $k = 0$, $V_m V_m^T g = g$ follows from (12). For the inductive step first note that $V_m H_m^k V_m^T = (V_m H_m V_m^T)^k$ for any integer k since $V_m^T V_m = I$. Then, assuming that the lemma is true for some $k < 0 \le m-2$, $(V_m H_m V_m^T)^k g = V_m H_m^k V_m^T g = L^k g$. Then,

$$\begin{aligned} V_m H_m^{k+1} V_m^T g &= V_m H_m V_m^T (V_m H_m V_m^T)^k g \\ &= V_m H_m V_m^T L^k g \text{ (Induction assumption)} \\ &= V_m V_m^T L V_m V_m^T L^k g \text{ (By (11))} \\ &= V_m V_m^T L^{k+1} g \text{ (By (13))} \\ &= L^{k+1} g \text{ (By (13) again.)} \end{aligned} \tag{15}$$

□

Now consider using the vector $g = g_n^{etd}$, to generate the subspace $\mathcal{K}_m(L, g_n^{etd})$, and the corresponding matrices H_m, V_m, by the Arnoldi algorithm. By Lemma 2.1 we have that, up to $k = m-1$,

$$V_m H_m^k V_m^T g_n^{etd} = L^k g_n^{etd}.$$

Thus, inserting the approximation $L^k \approx V_m H_m^k V_m^T$ in $\varphi_1(\Delta t L)$ the first m terms of the series definition (6) (from $k = 0$ to $k = m-1$) are correctly approximated. The Krylov approximation is then

$$\begin{aligned} \Delta t \varphi_1(\Delta t L) g_n \approx \Delta t \varphi_1(\Delta t V_m H_m V_m^T) g_n &= \Delta t V_m \varphi_1(\Delta t H_m) V_m^T g_n \\ &= ||g_n|| \Delta t V_m \varphi_1(\Delta t H_m) e_1. \end{aligned} \tag{16}$$

Let us introduce a shorthand notation for the Krylov approximation of the $\varphi-$function. Analogous to (4), for $\tau \in \mathbb{R}$ let

$$\tilde{p}_\tau \equiv \tau V_m \varphi_1(\tau H_m) V_m^T \approx p_\tau. \tag{17}$$

Using (16) and (17) we then approximate (5) by $u_{n+1}^{etd} = u_n^{etd} + \tilde{p}_{\Delta t} g_n^{etd}$. The key here is that the $\varphi_1(\Delta t H_m)$ now needs to be evaluated instead of $\varphi_1(\Delta t L)$. m is chosen such that $m \ll N$, and a classical method such as a rational Padé is used for $\varphi_1(\Delta t H_m)$, which would be prohibitively expensive for $\varphi_1(\Delta t L)$ for large N.

One step of the ETD1 scheme (2), under the approximation $\varphi_1(\Delta t L) \approx V_m \varphi_1(\Delta t H_m) V_m^T$, becomes

$$\begin{aligned} u_{n+1} &\approx u_n + \Delta t V_m \varphi_1(\Delta t H) V_m^T g_n \\ &= u_n ||g_n|| \Delta t V_m \varphi_1(\Delta t H) e_1, \end{aligned} \tag{18}$$

where e_1 is the vector in $\mathbb{R}^m$, where we have used the fact that g_n is orthonormal to all the columns of V_m except the first, by construction.

3. Recycling the Krylov subspace

In the Krylov subspace projection method described in Section 2, the subspace $\mathcal{K}_m(L, g_n)$ and thus the matrices H_m and V_m depend on g_n. At each step it is understood that a new subspace must be formed, and H_m, V_m be re-generated by the Arnoldi method, since g_n changes. In [26], it is demonstrated that splitting the timestep into two substeps, and recycling H_m and V_m, i.e. recycling the Krylov subspace, can be viable (in that it does not decrease the local order of the scheme, and apparently decreases the error). We expand on this concept with a more detailed analysis of the effect of this kind of recycled substepping applied to the locally second order ETD1 scheme (2) (EEM is considered in Appendix B). We replace a single step of length Δt of (18) with S substeps of length δt, such that $\Delta t = S\delta t$. We denote the approximations used in this scheme analogously to the notation for ETD1 earlier, without the *etd* superscript. Let us define the following notation for the substepping scheme,

$$u_{n+\frac{j}{S}} \approx u(t_n + j\delta t) \qquad \text{and} \qquad F_{n+\frac{j}{S}} \approx F(t_n + j\delta t, u_{n+\frac{j}{S}}).$$

that is, $u_{n+\frac{j}{S}}$ and $F_{n+\frac{j}{S}}$ are the approximations produced by the scheme for $u(t_n + j\delta t)$ and $F(t_n + j\delta t + jt, u_{n+\frac{j}{S}})$, respectively, at given discrete times. To clarify the subscript notation, $\frac{j}{S}$ denotes the jth substep, out of a total of S, during the nth complete step of the scheme. The n of course corresponds to the same nth whole step of ETD1

For $j = 1$ we calculate H_m, V_m, from g_n,

$$u_{n+\frac{1}{S}} = u_n + \delta t V_m \varphi_1(\delta t H_m) V_m^T g_n, \tag{19}$$

and for the remaining $S - 1$ steps,

$$u_{n+\frac{j}{S}} = u_{n+\frac{j-1}{S}} + \delta t V_m \varphi_1(\delta t H_m) V_m^T \left(L u_{n+\frac{j-1}{S}} + F_{n+\frac{j-1}{S}} \right), \; 1 < j \le S, \tag{20}$$

where the matrices H_m and V_m *are not re-calculated for any substep*, $j > 1$. We call substeps of the form (20) 'recycled steps' and substeps of the form (19) 'initial steps'.

Note that we could view (20) as approximating $L u_{n+\frac{j-1}{S}} + F_{n+\frac{j-1}{S}} = g_{n+\frac{j-1}{S}}$ by its orthogonal projection into $\mathcal{K}(L, g_n)$, i.e., $V_m V_m^T g_{n+\frac{j-1}{S}}$, such that,

$$\varphi_1(\delta t L) g_{n+\frac{j-1}{S}} \approx \varphi_1(\delta t L) V_m V_m^T g_{n+\frac{j-1}{S}} \approx V_m \varphi_1(\delta t H_m) V_m^T g_{n+\frac{j-1}{S}}.$$

The approximation to $u(t_n + \Delta t)$ at the end of the step of length Δt is then given by

$$u_{n+1} = u_{n+\frac{S-1}{S}} + \delta t V_m \varphi_1(\delta t H_m) V_m^T \left(L u_{n+\frac{S-1}{S}} + F_{n+\frac{S-1}{S}} \right). \tag{21}$$

The recycling steps (19), (20) can be succinctly expressed using the definition of $\tilde{p}_\tau$;

$$u_{n+\frac{1}{S}} = u_n + \tilde{p}_{\delta t} g_n, \tag{22}$$

$$u_{n+\frac{j}{S}} = u_{n+\frac{j-1}{S}} + \tilde{p}_{\delta t} \left(L u_{n+\frac{j-1}{S}} + F_{n+\frac{j-1}{S}} \right), \; 1 < j \le S. \tag{23}$$

We now make explicit the intended benefits of this scheme. Compared to ETD1, we are taking a regular step and adding several substeps. The regular step consists of two parts: (1) using the Arnoldi algorithm to generate H_m and V_m for the Krylov subspace, and (2) evolving the solution forward using (18). In practice the first step is much more expensive than the second. For the recycling scheme, we are adding $S - 1$ extra substeps that are comparable to part (2) in terms of cost (the first substep is essentially the ETD1 step with a reduced Δt). The intention then is that the substeps slightly increase the cost of each step, while at the same time increasing the accuracy of the scheme. The net effect is an improved efficiency - confirmed by experiments in Section 5. In Section 3.1, we derive an expression for the local error.

3.1. The local error of the recycling scheme

We now derive an expression for the local error of the scheme defined by (22), (23) and prove that the leading term decreases with the number of substeps S. Recall the standard definition of local error, (see for example [27, Section 9.5], or [28, Section 2.11–2.12]). The local error is the error which would be incurred by a scheme in a single step, if the data at the start of the step were exact, that is, we use the local error assumption that $u_n = u(t_n)$.

The local error can be thought as arising from two sources. First, the error of the method if it were possible to compute all matrix exponential functions (i.e. the φ-functions) exactly. This is the kind of error that is considered in, for example, [1] and is what is meant when we speak of say, the error of ETD1 being first order with respect to Δt and ETD2 being second order.

The second source of error comes from the practical reality of approximating the matrix exponential functions, by Krylov subspace, Leja point methods or others. See for example [23].

Consider ETD1; let the local error be

$$\text{local error of ETD} = \mathcal{E}_n + \mathcal{E}^{\mathcal{K}}_{n,m}$$

where $\mathcal{E}^{\mathcal{K}}_{n,m}$ is the Krylov approximation error such that,

$$\tilde{p}_\tau g_n = p_\tau g_n + \mathcal{E}^{\mathcal{K}}_{n,m}, \tag{24}$$

and $\mathcal{E}_n$ is the standard local error of the scheme, based on the assumption that φ- functions could be computed exactly. We make an important assumption about the accuracy of the initial Krylov approximation with respect to the error of the scheme.

Assumption 3.1. The parameters Δt, m, L, F are such that if $\mathcal{E}_n = O(\Delta t^a)$ and $\mathcal{E}^{\mathcal{K}}_{n,m} = O(\Delta t^b)$, then we assume $a < b$ so that we can write

$$\mathcal{E}_n + \mathcal{E}^{\mathcal{K}}_{n,m} = O(\Delta t^a).$$

That is, the local error is dominated by the contribution from $\mathcal{E}_n$ because $\mathcal{K}_{n,m}$ is much smaller as $\Delta t \to 0$.

Bounds on $\mathcal{K}^m_{n+1}$ can be found in for example [20,21]. Practically, we can always reduce Δt or increase m until Assumption 3.1 is satisfied. We will make use of Assumption 3.1 and investigate the non Krylov part of the local error of the recycling scheme, by deriving an expression for the deviation from the recycling scheme from ETD1, and thus the deviation of the local error from the local error of ETD1.

For the local error of the recycling scheme, the following result will be used.

Lemma 3.2. *For any $\tau_1, \tau_2 \in \mathbb{R}$, and any vector $v \in \mathbb{R}^N$,*

$$p_{\tau_1} v + p_{\tau_2}(L p_{\tau_1} v + v) = p_{\tau_1+\tau_2} v,$$

and the same relation holds for the Krylov approximations, that is,

$$\tilde{p}_{\tau_1} v + \tilde{p}_{\tau_2}(L \tilde{p}_{\tau_1} v + v) = \tilde{p}_{\tau_1+\tau_2} v.$$

Proof. We prove the second equation. The first can be proved using an almost identical argument, replacing $\tilde{p}_\tau$ by p_τ where appropriate.

By the definitions of $\tilde{p}_\tau$, and φ_1, i.e. $\tilde{p}_\tau = \tau V_m \varphi_1(\tau H_m) V_m^T = V_m H_m^{-1}\left(e^{\tau_2 H_m} - I\right) V_m^T$ we have

$$\tilde{p}_{\tau_2}(L\tilde{p}_{\tau_1} v + v) = V_m H_m^{-1}\left(e^{\tau_2 H_m} - I\right) V_m^T \left(L V_m H_m^{-1}\left(e^{\tau_1 H_m} - I\right) V_m^T + I\right) v.$$

After expanding the brackets and applying (9) this becomes $\tilde{p}_{\tau_2}\left(L\tilde{p}_{\tau_1} v + v\right) = V_m H_m^{-1}\left(e^{(\tau_2+\tau_1)H_m} - e^{\tau_1 H_m}\right) V_m^T v$. Now using the definition of $\tilde{p}_{\tau_1}$,

$$\begin{aligned}\tilde{p}_{\tau_1} v + \tilde{p}_{\tau_2}(L\tilde{p}_{\tau_1} v + v) &= V_m H_m^{-1}\left(e^{\tau_1 H_m} - I\right) V_m^T v + V_m H_m^{-1}\left(e^{(\tau_2+\tau_1)H_m} - e^{\tau_1 H_m}\right) V_m^T v \\ &= V_m H_m^{-1}\left(e^{(\tau_2+\tau_1)H_m} - I\right) V_m^T v,\end{aligned}$$

which is $\tilde{p}_{\tau_1+\tau_2} v$ as desired. □

Without recycling substeps, a single ETD1 step (2) of length Δt, using the polynomial Krylov approximation, would be:

$$u^{etd}_{n+1} = u^{etd}_n + \tilde{p}_{\Delta t} g^{etd}_n. \tag{25}$$

To examine the local error we compare u^{etd}_{n+1} with the u_{n+1} obtained after some number S of recycled substeps. We can write

$$u_{n+1} = u_n + \tilde{p}_{\Delta t} g_n + R^S_{n+1},$$

where R^S_{n+1} represents the deviation from (25) over one step. Then we have:

Lemma 3.3. *The approximation $u_{n+\frac{j}{S}}$ produced by j substeps of the recycling scheme (22), (23), satisfies*

$$u_{n+\frac{j}{S}} = u_n + \tilde{p}_{j\delta t} g_n + R^S_{n+\frac{j}{S}}, \tag{26}$$

with

$$R^S_{n+\frac{j}{S}} = \sum_{k=1}^{j} (I + \tilde{p}_{\delta t} L)^{j-k} \tilde{p}_{\delta t}(F_{n+\frac{k-1}{S}} - F_n). \tag{27}$$

Proof. By induction. For $j = 1$, $u_{n+\frac{1}{S}}$ is given by (22) and $R^S_{n+\frac{1}{S}} = 0$. Eq. (27) gives $R^S_{n+\frac{1}{S}} = \tilde{p}_{\delta t}(F_{n+\frac{0}{S}} - F_n) = 0$ as required.

Assume now (26) holds for some $j \geq 1$. Then $u_{n+\frac{j+1}{S}}$ is obtained by a step of (23). Using (26) we find,

$$u_{n+\frac{j+1}{S}} = u_{n+\frac{j}{S}} + \tilde{p}_{\delta t}\left(Lu_n + L\tilde{p}_{j\delta t} g_n + L R^S_{n+\frac{j}{S}} + F_{n+\frac{j}{S}}\right),$$

and since $Lu_n = g_n - F(t_n, u(t_n))$ (note the use of the local error assumption that $u_n = u(t_n)$), by the induction hypothesis we have,

$$\begin{aligned} u_{n+\frac{j+1}{S}} &= u_{n+\frac{j}{S}} + \tilde{p}_{\delta t}\left(g_n + L\tilde{p}_{\delta t}g_n + LR^S_{n+\frac{j}{S}} + F_{n+\frac{j}{S}} - F(t_n, u(t_n))\right) \\ &= u_{n+\frac{j}{S}} + \tilde{p}_{\delta t}(g_n + L\tilde{p}_{\delta t}g_n) + \tilde{p}_{\delta t}LR^S_{n+\frac{j}{S}} + \tilde{p}_{\delta t}(F_{n+\frac{j}{S}} - F_n) \\ &= u_n + \tilde{p}_{j\delta t}g_n + \tilde{p}_{\delta t}(g_n + L\tilde{p}_{\delta t}g_n) + (I + \tilde{p}_{\delta t}L)R^S_{n+\frac{j}{S}} + \tilde{p}_{\delta t}(F_{n+\frac{j}{S}} - F_n). \end{aligned}$$

Thus by Lemma 3.2 we have that,

$$u_{n+\frac{j+1}{S}} = u_n + \tilde{p}_{(j+1)\delta t}g_n + \tilde{p}_{\delta t}(F_{n+\frac{j}{S}} - F_n) + (I + \tilde{p}_{\delta t}L)R^S_{n+\frac{j}{S}}.$$

To complete the proof we need to show:

$$R^S_{n+\frac{j+1}{S}} = \tilde{p}_{\delta t}(F_{n+\frac{j}{S}} - F_n) + (I + \tilde{p}_{\delta t}L)R^S_{n+\frac{j}{S}}, \tag{28}$$

which we do now. By the induction hypothesis that (27) holds for j,

$$\begin{aligned} \tilde{p}_{\delta t}(F_{n+\frac{j}{S}} - F_n) + (I + \tilde{p}_{\delta t}L)R^S_{n+\frac{j}{S}} &= \tilde{p}_{\delta t}(F_{n+\frac{j}{S}} - F_n) + (I + \tilde{p}_{\delta t}L)\sum_{k=1}^{j}(I + \tilde{p}_{\delta t}L)^{j-k}\tilde{p}_{\delta t}(F_{n+\frac{j-1}{S}} - F_n) \\ &= \sum_{k=1}^{j+1}(I + \tilde{p}_{\delta t}L)^{j+1-k}\tilde{p}_{\delta t}(F_{n+\frac{k-1}{S}} - F_n) = R^S_{n+\frac{j+1}{S}}. \end{aligned} \tag{29}$$

Hence the lemma is proved. □

Using (26) we now express the leading order term of the local error in terms of S. First we examine the leading order term of R^S_{n+1}.

Assumption 3.4. Assume that the recycling scheme defined by (22) and (23) has been at least first order accurate up to step n such that we have $u_n = u(t_n) + O(\Delta t)$ and thus we can write $g_n = g(t_n, u(t_n)) + O(\Delta t)$ or equivalently $g_n = g(t_n, u(t_n)) + O(\delta t)$ after a Taylor expansion (noting that $\Delta t = S\delta t$, we can write $O(\Delta t^a)$ as $O(\delta t^a)$ for any integer a).

This assumption is justified as follows. For $n = 0$ we have that $u_n = u(t_n)$ when the initial data is exact (we assume here that it is). In the following we will prove, using Lemma 3.5 and then lemmas which depend on it, that the recycling scheme is indeed first order for any n. The assumption is therefore proved inductively.

Before stating the lemma and its proof we clarify some notation. Let the total derivative of F with respect to time be

$$\frac{dF}{dt}(t, u(t)),$$

while partial derivatives with respect to time and u are, respectively,

$$\frac{\partial F}{\partial t}(t, u(t)); \frac{\partial F}{\partial u}(t, u(t)).$$

The standard relation between partial and total derivatives applies (dropping the brackets for brevity):

$$\frac{dF}{dt} = \frac{\partial F}{\partial t} + \frac{\partial F}{\partial u}\frac{\partial u}{\partial t}.$$

Note that $\frac{\partial F}{\partial u}$ is a Jacobian matrix, and that since u depends only on t we may write $\frac{\partial u}{\partial t}$ as $\frac{du}{dt}$ or simply $\frac{\partial u}{\partial t} = g$, given the definition of g. The resulting relation $\frac{dF}{dt} = \frac{\partial F}{\partial t} + \frac{\partial F}{\partial u}g$ will be used shortly.

Further it is useful to clarify the Taylor expansions of $\tilde{p}_{j\delta t}$. Combining the definitions (6) and (17), we see that

$$\tilde{p}_{j\delta t} = j\delta t V_m \varphi_1(j\delta t H_m)V_m^T = V_m \sum_{k=0}^{\infty}\frac{(j\delta t H_m)^k}{(k+1)!}V_m^T.$$

Looking only at the leading term this gives us,

$$\tilde{p}_{j\delta t} = j\delta t V_m V_m^T + O(\delta t^2),$$

The action of the matrix $\tilde{p}_{j\delta t}$ on the vector g_n is then,

$$\tilde{p}_{j\delta t}g_n = j\delta t g_n + O(\delta t^2), \tag{30}$$

because $V_m V_m^T g_n = g_n$, as g_n is in the Krylov subspace associated with V_m. Eq. (30) will be used in the proof of the lemma, which we now state.

Lemma 3.5. *Assume that the function $F(t, u(t))$ is such that its total derivative with respect to time $\frac{dF}{dt}(t, u(t))$ exists. Also assume that Assumption 3.4 holds. Then, the term $R^S_{n+\frac{j}{S}}$ in Lemma 3.3, when expanded in powers of Δt, satisfies*

$$R^S_{n+\frac{j}{S}} = \frac{j(j-1)}{2}\delta t^2 V_m V_m^T \frac{dF}{dt}(t_n, u_n) + O(\Delta t^3). \tag{31}$$

Proof. By induction. For the case $j = 1$, we see from (27) that $R^S_{n+\frac{1}{S}} = 0$ since $(F_{n+\frac{0}{S}} - F_n) = 0$. Thus (31) is true for $j = 1$. Now assume the result holds for some j.

Observe that from (26), the induction assumption that $R^S_{n+\frac{j}{S}} = O(\delta t^2)$, and (30), we have that $u_{n+\frac{j}{S}} = u_n + j\delta t g_n + O(\delta t^2)$. Then we can express the term $F_{n+\frac{j}{S}}$ follows:

$$\begin{aligned} F(t_{n+\frac{j}{S}}, u_{n+\frac{j}{S}}) &= F(t_{n+\frac{j}{S}}, u_n + (j\delta t g_n + O(\delta t^2))) \\ &= F(t_n, u_n) + j\delta t \frac{\partial F}{\partial t}(t_n, u_n) + j\delta t \frac{\partial F}{\partial u}(t_n, u_n) g_n + O(\delta t^2) \\ &= F(t_n, u_n) + j\delta t \frac{dF}{dt}(t_n, u_n) + O(\delta t^2). \end{aligned}$$

Note that for the final step we have used that $g_n = g(t_n, u(t_n)) + O(\delta t) = \frac{du}{dt}(t_n) + O(\delta t)$; using Assumption 3.4, the assumptions of the lemma and (1) and the definition of $g(t)$.

We thus have that

$$(F_{n+\frac{j}{S}} - F_n) = j\delta t \frac{dF}{dt}(t_n, u_n) + O(\delta t^2). \tag{32}$$

We then insert (32) into the inductive expression (28) for $R^S_{n+\frac{j}{S}}$ and use the expansion $\tilde{p}_{\delta t} = \delta t V_m V_m^T + O(\delta t^2)$ to get

$$R^S_{n+\frac{j+1}{S}} = \delta t V_m V_m^T j\delta t \frac{dF}{dt}(t_n, u_n) + (I + \delta t V_m V_m^T L) R^S_{n+\frac{j}{S}} + O(\delta t^3).$$

Using the induction assumption (31),

$$R^S_{n+\frac{j+1}{S}} = \delta t V_m V_m^T j\delta t \frac{dF}{dt}(t_n, u_n) + \frac{j(j-1)}{2}\delta t^2 V_m V_m^T \frac{dF}{dt}(t_n, u_n) + O(\Delta t^3).$$

Noting that $\Delta t = S\delta t$ we can write $O(\Delta t^3)$ as $O(\delta t^3)$. Collecting terms we have,

$$R^S_{n+\frac{j+1}{S}} = \left(\frac{j(j-1)}{2} + j\right)\delta t^2 V_m V_m^T \frac{dF}{dt}(t_n, u_n) + O(\Delta t^3).$$

The lemma follows since $\frac{j(j-1)}{2} + j = \frac{j(j+1)}{2}$. □

The leading local error term of the ETD1 scheme without substeps is well known to be $\frac{\Delta t^2}{2}\frac{dF}{dt}(t)$ (see [29]), so that we can finally recover the leading term from Lemma 3.3.

Corollary 3.6. *The leading term of the recycling scheme after j steps is*

$$u_{n+\frac{j}{S}} = u_n + j\delta t g_n + \frac{j^2\delta t^2}{2} L g_n + \frac{j(j-1)}{2}\delta t^2 V_m V_m^T \frac{dF}{dt}(t_n, u_n) + O(\delta t^3). \tag{33}$$

Corollary 3.7. *The local error $u(t_n + \Delta t) - u_{n+1}$ of an ETD1 Krylov recycling scheme is second order for any number S of recycled substeps. Moreover, the local error after j recycled steps is*

$$u(t_n + j\delta t) - u_{n+\frac{j}{S}} = \frac{(j\delta t)^2}{2}\left(I - \frac{j-1}{j} V_m V_m^T\right)\frac{dF}{dt}(t) + O(\delta t^3).$$

In particular

$$u(t_n + \Delta t) - u_{n+1} = \frac{\delta t^2}{2}\left(S^2 - S(S-1) V_m V_m^T\right)\frac{dF}{dt}(t) + O(\delta t^2), \tag{34}$$

or in terms of Δt

$$u(t_n + \Delta t) - u_{n+1} = \frac{\Delta t^2}{2}\left(I - \frac{S-1}{S} V_m V_m^T\right)\frac{dF}{dt}(t) + O(\Delta t^2). \tag{35}$$

It is interesting to compare (35) with the leading term of the local error of regular ETD1, $\frac{\Delta t^2}{2}\frac{dF}{dt}(t)$. Since $V_m V_m^T$ is the orthogonal projector into $\mathcal{K}$, then we can see that the $\frac{\Delta t^2}{2}\frac{S-1}{S} V_m V_m^T \frac{dF}{dt}(t)$ part in (35) is the projection of the ETD1 error into $\mathcal{K}$, multiplied by a factor $\frac{S-1}{S} \leq 1$. Thus, in the leading term, according to (35), the recycling scheme reduces the error of

ETD1 by effectively eliminating the part of the error which lives in $\mathcal{K}$. In the limit $S \to \infty$, the entirety of the error in $\mathcal{K}$ will be eliminated. The effectiveness of the recycling scheme therefore depends on how much of $\frac{dF}{dt}(t)$ can be found in $\mathcal{K}$.

Corollary 3.7 shows that using $S > 1$ recycled substeps is advantageous over the basic ETD1 scheme, in the sense of reducing the magnitude of the leading local error term, whenever

$$\left|\left|\left(I - \frac{S-1}{S} V_m V_m^T\right)\frac{dF}{dt}(t)\right|\right| < \left|\left|\frac{dF}{dt}(t)\right|\right|, \tag{36}$$

where $||\cdot||$ is a given vector norm. We show in Lemma 3.9 that increasing S will decrease the Euclidean norm $||\cdot||_2$ of the leading term of the local error. First we require a result on $V_m V_m^T$, the projector into the Krylov Subspace $\mathcal{K}$.

Remark 3.8. Let $x \neq 0$ be a vector such that $V_m V_m^T x \neq 0$, then for $\alpha \in \mathbb{R}$

$$\left|\left|\left(I - \alpha V_m V_m^T\right)x\right|\right|_2^2 = ||x||_2^2 + [(1-\alpha)^2 - 1]\left|\left|V_m V_m^T x\right|\right|_2^2. \tag{37}$$

Proof. An elementary result for orthogonal projectors (see, e.g. [30]) is that

$$||x||_2^2 = \left|\left|V_m V_m^T x\right|\right|_2^2 + \left|\left|(I - V_m V_m^T)x\right|\right|_2^2, \tag{38}$$

which follows from $V_m V_m^T x \perp (I - V_m V_m^T)x$ (the orthogonality of $V_m V_m^T x$ and $(I - V_m V_m^T)x$) and the definition of the Euclidean norm. Eq. (37) is a generalisation of (38) as can be shown as follows.

Write $x - \alpha V_m V_m^T x = (I - V_m V_m^T)x + (1-\alpha)V_m V_m^T x$, and then, noting that $(I - V_m V_m^T)x \perp (1-\alpha)V_m V_m^T x$, we see that

$$\left|\left|x - \alpha V_m V_m^T x\right|\right|_2^2 = \left|\left|(I - V_m V_m^T)x\right|\right|_2^2 + (1-\alpha)^2\left|\left|V_m V_m^T x\right|\right|_2^2.$$

Using (38) to substitute for $\left|\left|(I - V_m V_m^T)x\right|\right|_2^2$ yields (37). □

Lemma 3.9. *Assume $\frac{dF}{dt}(t) \neq 0$ and $V_m V_m^T \frac{dF}{dt}(t) \neq 0$. Let E_{S_1} be the local error using the recycling scheme over a timestep of length Δt with $S_1 \geq 1$ substeps, and E_{S_2} the local error with S_2 substeps with $S_2 > S_1$. Then,*

$$||E_{S_2}||_2 < ||E_{S_1}||_2.$$

Proof. The local errors E_{S_k}, $k = 1, 2$ are given in Corollary 3.7. Let $\frac{S_k - 1}{S_k} \equiv \beta_k$, $k = 1, 2$. We need to show that

$$\left|\left|\left(I - \beta_2 V_m V_m^T\right)\frac{dF}{dt}(t)\right|\right|_2 < \left|\left|\left(I - \beta_1 V_m V_m^T\right)\frac{dF}{dt}(t)\right|\right|_2.$$

Let $x \equiv (I - \beta_1 V_m V_m^T)\frac{dF}{dt}(t)$, then $(I - \beta_2 V_m V_m^T)\frac{dF}{dt}(t) = x - (\frac{\beta_1 - \beta_2}{\beta_1 - 1})V_m V_m^T x$ (showing this involves using $V_m V_m^T V_m V_m^T = V_m V_m^T$). Letting $\gamma \equiv \frac{\beta_1 - \beta_2}{\beta_1 - 1}$, we then need to show

$$\left|\left|\left(I - \gamma V_m V_m^T\right)x\right|\right|_2 < ||x||_2. \tag{39}$$

Note that we have that $V_m V_m^T x \neq 0$ from the assumptions. This is because,

$$V_m V_m^T x = \left(V_m V_m^T - \beta_1 V_m V_m^T\right)\frac{dF}{dt}(t),$$

since $V_m V_m^T V_m V_m^T \frac{dF}{dt}(t) = V_m V_m^T \frac{dF}{dt}(t)$, as $V_m V_m^T \frac{dF}{dt}(t)$ is already entirely within $\mathcal{K}$. Then,

$$V_m V_m^T x = (1 - \beta_1) V_m V_m^T \frac{dF}{dt}(t).$$

We have that $1 - \beta_1 = \frac{1}{S_1} \neq 0$ and $V_m V_m^T \frac{dF}{dt}(t) \neq 0$, so that $V_m V_m^T x \neq 0$.

To prove the lemma we apply (37) to x, with γ in place of α. If we have that $[(1-\gamma)^2 - 1] < 0$, then (39) is true since $V_m V_m^T x \neq 0$. An equivalent requirement is $\gamma \in (0, 2)$. Some algebra gives us $\gamma = 1 - \frac{S_1}{S_2}$. Since $S_2 > S_1$, it follows that $\gamma \in (0, 2)$. □

From Lemma (3.9), we see that S recycled Krylov substeps not only maintains the local error order of the ETD1 scheme, but also decreases the 2-norm of the leading term with increasing S. Note that the leading term does not tend towards zero as $S \to \infty$, but towards a constant. We thus expect diminishing returns in the increase in accuracy with increasing S, and the existence of an optimal S for efficiency.

4. Using the additional substeps for correctors

We now establish a new second order scheme based on a finite difference approximation to the derivative of the non-linear term $F(t)$ and the recycling scheme given in (19) and (20).

The first step is to expand the local error for the standard ETD1 scheme. Using variation of constants and a Taylor series expansion of $F(t, u(t))$, the exact solution of (1) can be expressed as a power series (see for example [5,29])

$$u(t_n + \Delta t) = e^{\Delta t L} u(t_n) + \sum_{k=1}^{\infty} \Delta t^k \varphi_k(\Delta t L) F^{(k-1)}(t_n, u_n) + O(\Delta t^k), \tag{40}$$

with $F^{(k)}(t_n, u_n) = \frac{d^k F}{dt^k}(t_n, u_n)$. Under the local error assumption $u_n = u(t_n)$, the local error of the ETD1 step given in (2) is

$$E^{etd}_{n+1} \equiv u(t_n + \Delta t) - u^{etd}_n + p_{\Delta t} g_n = \sum_{k=2}^{\infty} \Delta t^k \varphi_k(\Delta t L) F^{(k-1)}(t_n). \tag{41}$$

Since the approximation from a substepping scheme is related to the approximation from the ETD1 scheme (over one step) by $u_{n+1} = u^{etd}_{n+1} + R^S_{n+1}$, we have the local error for the recycling scheme:

$$u(t_n + \Delta t) - u_n = E^{etd}_{n+1} - R^S_{n+1}. \tag{42}$$

The terms of error expression (42) at arbitrary order can be found using (40), Lemma 3.3, and the information on Krylov projection methods in Section 2. We see that the expansion consists of terms involving the value of $F(t)$ or derivatives thereof at various substeps. These terms can be approximated by finite differences of the values for F at the different substeps, and used as a corrector to eliminate terms for the error.

We consider extrapolation in the leading error in the case of two substeps, that is $S \equiv 2$. Assume that the error from the Krylov approximation, $\mathcal{E}^K_{n,m}$, is negligible compared to E_n and R^S_{n+1}, so that it does not introduce any terms at the first and second and third order expansion of E_n and R^S_{n+1}. Then we can express exactly the leading second and third order error terms.

First we have the leading terms of E^{etd}_n from (41),

$$\begin{aligned} E^{etd}_n &= \Delta t^2 \varphi_2(\Delta t L) F^{(1)}(t_n, u_n) + \Delta t^3 \varphi_3(\Delta t L) F^{(2)}(t_n, u_n) + O(\Delta t^4) \\ &= \frac{\Delta t^2}{2!} F^{(1)} + \frac{\Delta t^3 L}{3!} F^{(1)} + \frac{\Delta t^3}{3!} F^{(2)} + O(\Delta t^4). \end{aligned} \tag{43}$$

We also have the leading terms of R^2_{n+1} (from two substeps, recall (27))

$$\begin{aligned} R^2_{n+1} &= \tilde{p}_{\frac{\Delta t}{2}}(F_{n+\frac{1}{2}} - F_n) \\ &= \frac{\Delta t}{2} V_m \left(I + \frac{\Delta t H_m}{2} + \dots \right) V_m^T (F_{n+\frac{1}{2}} - F_n) \\ &= \frac{\Delta t}{2} V_m V_m^T (F_{n+\frac{1}{2}} - F_n) + V_m \frac{\Delta t^2 H_m}{4} V_m^T (F_{n+\frac{1}{2}} - F_n) + O(\Delta t^3). \end{aligned} \tag{44}$$

Note that the terms in (44) are an order higher than written since $F_{n+\frac{1}{2}} - F_n = \frac{\Delta t}{2} F^{(1)}(t_n, u_n) + O(\Delta t^2)$. We then have that

$$\begin{aligned} u(t_n + \Delta t) = u_{n+1} &+ \frac{\Delta t^2}{2!} F^{(1)} - \frac{\Delta t}{2} V_m V_m^T (F_{n+\frac{1}{2}} - F_n) \\ &+ \frac{\Delta t^3 L}{3!} F^{(1)} + \frac{\Delta t^3}{3!} F^{(2)} - V_m \frac{\Delta t^2 H_m}{4} V_m^T (F_{n+\frac{1}{2}} - F_n) + O(\Delta t^4). \end{aligned} \tag{45}$$

The idea now is as follows. Define a corrected approximation:

$$u^{(c)}_{n+1} \equiv u_{n+1} + C - \frac{\Delta t}{2} V_m V_m^T (F_{n+\frac{1}{2}} - F_n). \tag{46}$$

In (46), C is a corrector intended to cancel out some of the leading terms in (45). The term $\frac{\Delta t}{2} V_m V_m^T (F_{n+\frac{1}{2}} - F_n)$ is the only leading term in (45) to involve the matrix V_m, and so is added directly to the corrected approximation (46) to allow C to be free of dependence on the matrix V_m. Indeed, C will be a linear combination of the the three function values of $F(t)$, F_n, $F_{n+\frac{1}{2}}$ and F_{n+1}, available at the end of the full step. The approximation to u produced by substeps of the scheme, and thus also to F, is locally second order. We define the C term as follows, with coefficients α, β, γ to be chosen later.

$$\begin{aligned} C &\equiv \Delta t \alpha F_n + \Delta t \beta F_{n+\frac{1}{2}} + \Delta t \gamma F_{n+1} \\ &= \Delta t \alpha F(t_n) + \Delta t \beta F\left(t_n + \frac{\Delta t}{2} \right) + \Delta t \gamma F(t_n + \Delta t) + \Delta t^3 E_c + O(\Delta t^4) \\ &= \Delta t (\alpha + \beta + \gamma) F + \Delta t^2 \left(\frac{\beta}{2} + \gamma \right) F^{(1)} + \frac{\Delta t^3}{2} \left(\frac{\beta}{4} + \gamma \right) F^{(2)} + \Delta t^3 E_c + O(\Delta t^4) \end{aligned} \tag{47}$$

where we have used that $F_n = F(t_n, u(t_n))$ (under the local error assumptions), $F_{n+\frac{1}{2}} = F\left(t_n + \frac{\Delta t}{2}\right) + O(\Delta t^2)$ and so on. The new term $\Delta t^3 E_c$ is introduced to represent the $O(\Delta t^3)$ error in writing $\Delta t F_{n+\frac{1}{2}}$ as $\Delta t F\left(t_n + \frac{\Delta t}{2}\right)$, and so on.

From (47), we must choose the coefficients to satisfy the two conditions

$$\alpha + \beta + \gamma = 0, \qquad \text{and} \qquad \frac{\beta}{2} + \gamma = \frac{1}{2}.$$

With these values of the parameters, the local error of the corrected approximation is

$$\begin{aligned} u(t_n + \Delta t) - u_{n+1}^{(c)} &= \frac{\Delta t^3}{3!} F^{(2)} - V_m \frac{\Delta t^2 H_m}{4} V_m^T (F_{n+\frac{1}{2}} - F_n) + \frac{\Delta t^3}{2}\left(\frac{\beta}{4} + \gamma\right) F^{(2)} - \Delta t^3 E_c + O(\Delta t^4) \\ &= \frac{\Delta t^3}{3!} F^{(2)} - V_m \frac{\Delta t^2 H_m}{8} V_m^T F^{(1)} + \frac{\Delta t^3}{2}\left(\frac{\beta}{4} + \gamma\right) F^{(2)} - \Delta t^3 E_c + O(\Delta t^4). \end{aligned} \tag{48}$$

We have three coefficients to determine, and two constraints. We are therefore in a position to pick another constraint to reduce the new leading error in (48). It would be helpful to know the form of the error term E_c, introduced by the approximation of F in (47). We have:

$$\begin{aligned} F_{n+\frac{1}{2}} &= F\left(t_{n+\frac{1}{2}}, u\left(t_{n+\frac{1}{2}}\right) - \frac{\Delta t^2}{8} F' + O(\Delta t^3)\right) \\ &= F\left(t_{n+\frac{1}{2}}, u\left(t_{n+\frac{1}{2}}\right)\right) - \frac{\Delta t^2}{8} \frac{\partial F}{\partial u} F' + O(\Delta t^3), \end{aligned} \tag{49}$$

using Corollary 3.7. We also have

$$\begin{aligned} F_{n+\frac{2}{2}} &= F\left(t_{n+1}, u(t_{n+1}) - \frac{\Delta t^2}{2}\left(I - \frac{1}{2} V_m V_m^T\right) \frac{dF(t)}{dt} + O(\Delta t^3)\right) \\ &= F(t_{n+1}, u(t_{n+1})) - \frac{\Delta t^2}{2} \frac{\partial F}{\partial u}\left(I - \frac{1}{2} V_m V_m^T\right) \frac{dF(t_n)}{dt} + O(\Delta t^3), \end{aligned} \tag{50}$$

E_c is then

$$-\beta \frac{1}{8} \frac{\partial F}{\partial u} \frac{dF(t_n)}{dt} - \gamma \frac{1}{2} \frac{\partial F}{\partial u}\left(I - \frac{1}{2} V_m V_m^T\right) \frac{dF(t_n)}{dt}.$$

Substituting into (48) ,

$$\begin{aligned} u(t_n + \Delta t) - u_{n+1}^{(c)} =& \frac{\Delta t^3}{3!} F^{(2)} - \frac{\Delta t^2}{4} V_m H_m V_m^T (F_{n+\frac{1}{2}} - F_n) - \frac{\Delta t^3}{2}\left(\frac{\beta}{4} + \gamma\right) F^{(2)} \\ &- \Delta t^3 \frac{\partial F}{\partial u}\left(\left(\frac{\beta}{8} + \frac{\gamma}{2}\right) I - \frac{\gamma}{4} V_m V_m^T\right) \frac{dF(t_n)}{dt}. \end{aligned} \tag{51}$$

We have the option here to use the final constraint to eliminate the coefficient of $F^{(2)}$ in the leading term:

$$\frac{\Delta t^3}{3!} - \frac{\Delta t^3}{2}\left(\frac{\beta}{4} + \gamma\right) = 0.$$

Note that E_c cannot be eliminated without taking the inverse of $V_m V_m^T$, so this is not an efficient option. It can be seen that the values that satisfy the three constraints are:

$$\alpha = -\frac{5}{6}, \quad \beta = \frac{2}{3}, \quad \gamma = \frac{1}{6}.$$

Of course E_c also depends on the values of α, β, γ, so the magnitude of the third order term will be affected by the choice of these values also through E_c. With the choices given above, we have the numerical scheme

$$u_{n+1}^{(c)} = u_{n+1} + C - \frac{\Delta t}{2} V_m V_m^T (F_{n+\frac{1}{2}} - F_n); \tag{52}$$

that is,

$$\begin{aligned} u_{n+1}^{(c)} =& u_{n+1} - \Delta t \frac{5}{6} F_n + \Delta t \frac{2}{3} F_{n+\frac{1}{2}} + \Delta t \frac{1}{6} F_{n+1} \\ &- \frac{\Delta t}{2} V_m V_m^T (F_{n+\frac{1}{2}} - F_n). \end{aligned} \tag{53}$$

and the E_c term in (51) becomes:

$$-\Delta t^3 \frac{\partial F}{\partial u}\left(\frac{2}{12} I - \frac{1}{24} V_m V_m^T\right) \frac{dF(t_n)}{dt}.$$

Here we have used all the extra information from the two substeps to completely eliminate the lowest order from the local error, and a part of the new leading order term for the scheme. A more thorough use of the error expressions in the lemmas here may give rise to recycling schemes that use more substeps and are able to completely eliminate higher order terms from the error, leading to a kind of new exponential Runge-Kutta framework involving recycled Krylov subspaces. Below we demonstrate the efficacy of our two-step corrected recycling scheme with numerical examples. In Appendix B we show how to apply the analysis of the substepping method to the locally third order exponential integrator scheme EEM.

5. Numerical results

Here we examine the performance of the recyling scheme (22), (23) and the corrector scheme (52) (for the first two examples). All the schemes were implemented and tested in Matlab. We provide Matlab code for the first order scheme in Appendix A.

The PDEs investigated in these experiments are all advection-diffusion-reaction equations, which are converted into semi-linear ODE systems (1) by spatial discretisation before our timestepping schemes are applied (see for example, [25] for more details). We use the notation $U(\mathbf{x}, t) \in \mathbb{R}$ to represent the solution to the PDE, while $u(t) \in \mathbb{R}^N$ represents the solution of the corresponding ODE system. The spatial discretisation is a simple finite volume method in all the examples. In examples 2 and 3, the grid was using code from MRST [31]. We compare the second order corrector scheme (52) to both the standard second order exponential integrator (ETD2; refer to for example Eq. (6) in [1]) and standard second order exponential Rosenbrock scheme (ROS2; the same as the simplest exponential Rosenbrock scheme described in [3,4]). For the first two experiments, the error is estimated by comparison with a low Δt comparison solve u_{comp} with ETD2.

We state these two schemes for reference. ETD2 is,

$$u_{n+1} = \phi_0(\Delta t L)u_n + \Delta t\phi_1(\Delta t L)F_n + \Delta t\phi_2(\Delta t L)(F_n - F_{n-1}),$$

a multistep scheme. The scheme ROS2 is,

$$u_{n+1} = u_n + \Delta t\phi_1\left(\Delta t\frac{\partial g_n}{\partial u}\right)g_n,$$

where $\frac{\partial g_n}{\partial u} = L + \frac{\partial F_n}{\partial u}$ is a Jacobian.

Our ETD2 and ROS2 implementations use phipm.m [23] for each timestep. The function phipm is amongst the best Krylov subspace based approximation algorithm for exponential integrators we are aware of; it uses an adaptive substepping algorithm to compute approximations to linear combinations of the action of φ-functions on fixed vectors, and can provide high efficiency by dynamically altering the substep length and the Krylov subspace dimension m (for a new subspace is computed each step), at each substep. For example, consider ETD1, and recall that we start a step n of that scheme with data u_n^{etd} and F_n^{etd}. We could then use phipm to approximate $\varphi_0(\Delta t L)u_n^{etd} + \varphi_1(\Delta t L)F_n^{etd}$ required to advance to step $n+1$ (observe that this is equivalent to (2)). The function automatically determines how many substeps to use and which Krylov subspace dimension to use in each substep.

It is important to note the differences between phipm and our method. Both are based on Krylov subspace approximation techniques and both use substepping, but have different goals. Phipm uses multiple Krylov subspaces in order to optimise ballancing between minimising the error in approximating the φ- functions and cost, and takes fixed vectors as input. This is sufficient for implementing schemes such as ETD1/2 or ROS2, where the nonlinearity $F(t)$ is only calculated once per timestep. By contrast, our methods are based around updating the $F(t)$ every supstep - the method we have presented is a variation on ETD1, designed to have a reduced local error, and not a method for imple- menting ETD1 with minimal Krylov error. We make use of Assumption 3.1; as a result of which we have to use m sufficient to keep the Krylov error sufficiently small. We have chosen to use phipm in our implementations of ETD2 and ROS2 for comparison as it represents a best of breed of existing modern implementations for efficiently approximating φ-functions. We note that our our experiments indicate the 2-step extrapolation scheme nonethelss exhibits com- parable or better efficiency compared to the comparison schemes, even though we use values of m that may seem quite high in order to ensure Assumption 3.1.

For the comparison schemes ETD2 and ROS2, phipm requires the following parameters: an initial Krylov subspace dimension m, and an error tolerance. For our comparison solve runs, the values $m = 30$ and 10^{-7} were chosen to be sufficiently accurate respectively for these parameters. The first two experiments are also found in [25]; see this for more details. For the third experiment a comparison solve was prohibitively time consuming, so error was instead estimated by differencing successive results. That is, let $u[\Delta t]$ be the approximation produced by a scheme with constant timestep Δt. Assuming that the scheme is globally first order, and neglecting the higher order terms of the error, then,

$$u[\Delta t] \approx u(T) + \epsilon\Delta t,$$

where ϵ is the coefficient of the leading term of the error for the scheme, and $u(T)$ is the true solution. In particular, we use timesteps differing by a factor of two, and since,

$$u[2\Delta t] \approx u(T) + 2\epsilon\Delta t,$$

then the difference

$$u[2\Delta t] - u[\Delta t] \approx \epsilon\Delta t$$

is an approximation for the error for the scheme $u[\Delta t]$. The norm is then taken on this value.

For every experiment we estimate the error in a discrete aproximation of the $L^2(\Omega)$ norm, where Ω is the computational domain.

For timing the schemes we used Matlab's Tic and Toc functions, therefore the units for time are in seconds.

5.1. Allen–Cahn type reaction diffusion

We approximate the solution to the PDE,

$$\frac{dv}{dt} = \nabla^2 Dv + v - v^3.$$

The (1D) spatial domain for this experiment was $\Omega = [0, 100]$ This was discretised into a grid of $N = 100$ cells. We imposed no flow boundary conditions, i.e., $\frac{\partial u}{\partial x} = 0$ where $x = 0$ or $x = 100$. There was a uniform diffusivity field of $D(x) = 1.0$. The initial condition was $u(x, 0) = \cos\left(\frac{2\pi x}{N}\right)$ and we solved to a final time $T = 1.0$.

In Fig. B.1 a and c, we show the estimated error against Δt, for the recycling scheme with varying number of substeps S, ($S = 1, 2, 5, 10, 50, 100$). Note that $S = 1$ is the standard ETD1 integtrator. The behaviour is as expected; increasing S decreases the error and the scheme is first order. The diminishing returns of increasing S (see (35)) can also be observed; for example compare the significant increase in accuracy in increasing S from 1 to 5, with the lesser increase in accuracy in increasing S from 5 to 10. Fig. B.1 shows this more emphatically - the increase in accuracy in increasing S from 10 to 50 is significant, but the effect of increasing S from 50 to 100 is very small. The limiting value of the error with respect to S discussed above is clearly close to being reached here.

In Fig. B.1 b and d we plot estimated error against cputime to demonstrate the efficiency of the scheme with varying S. In this case increasing S appears to increase efficiency until an optimal S is reached, after which it decreases, as predicted. Fig. B.1 d shows that the optimal S lies between 50 and 100 for this system.

In Fig. B.2, we examine the 2-step corrector (52). Plot (a) shows estimated error against Δt. The corrector scheme is second order as intended, and has quite high accuracy compared to the other two schemes, possibly due to the heuristic attempt to decrease the error in the leading term (see discussion in Section 4). In plot (b) we see that the 2-step corrector is of comparable efficiency to ROS2.

In Fig. B.1 b, we see that for the same cputime, increasing S from 1 to 10 decreases the estimated error by roughly an order of magnitude. We can see in Fig. B.1 d that increasing S from 10 to 50 can further decrease error for a fixed cputime, though less significantly. Comparing a fixed cputime in Fig. B.1 b and Fig. B.2 b indicates that the second order, 2-step corrector method can produce error more than one order of magnitude smaller than the first order recycling scheme with $S = 10$.

In Fig. B.3, we show an alternative measure of the efficiency, plotting the logarithm of error per unit time is plotted against the logarithm of S. Each curve is a different fixed timestep Δt value. Minima in the curves would indicate an optimal S for efficiency, although in this experiment this is not reached for within the range of S used - increasing S continues to improve the efficiency measure up to and possibly beyond $S = 100$. We can observe the value decreasing less rapidly as S increases, demonstrating the predicted diminishing returns.

5.2. Fracture system with Langmuir-type reaction

We approximate the solution to the PDE,

$$\frac{dv}{dt} = \nabla \cdot (\nabla Dv + Vv) - \frac{0.02}{D(\mathbf{x})^2}\frac{v}{1+v}. \tag{54}$$

where $D(\mathbf{x})$ is the diffusivity and $V(\mathbf{x})$ is the velocity. In this example a single layer of cells is used, making the problem effectively two dimensional. The domain is $\Omega = 10 \times 10 \times 10$ metres, divided into $100 \times 100 \times 1$ cells of equal size. We impose no-flow boundary conditions on every edge. The initial condition imposed is initial $v(\mathbf{x}) = 0$ everywhere except at $\mathbf{x} = (4.95, 9.95)^T$ where $v(\mathbf{x}) = 1$. The diffusivity D in the grid varies with $\mathbf{x}$, in a way intended to model a fracture in the medium. A subset of the cells in the 2D grid were chosen to be cells in the fracture. These cells were chosen by a weighted random walk through the grid (weighted to favour moving in the positive y-direction so that the fracture would bisect the domain). This process started on an initial cell which was marked as being in the fracture, then randomly chose a neighbour of the cell and repeated the process. We set the diffusivity to be $D = 100$ on the fracture and $D = 0.1$ elsewhere. There is also a constant velocity V field in the system, uniformly one in the x-direction and zero in the other directions in the domain, i.e., $\mathbf{v}(\mathbf{x}) = (1, 0, 0)^T$, to the right in Fig. B.5. The initial condition was $c(\mathbf{x}) = 0$ everywhere except at $\mathbf{x} = (4.95, 9.95)^T$ where $c(\mathbf{x}) = 1$.

In Fig. B.5, we show the final state of the system at time $T = 2.4$. The result in plot a) was produced with the 2-step recycling scheme with a timestep $\Delta t = 2.4 \times 10^{-4}$. Plot b) shows the high accuracy comparison ETD2 solve, produced with $\Delta t = 2.4 \times 10^{-5}$.

In Fig. B.6, we demonstrate the effect of increasing the number of substeps S on the error. Fig. B.6 a shows estimated error against timestep Δt, for schemes using $S = 1, 2, 5, 10$ substeps, while Fig. B.6 c shows the same for schemes using $S = 10, 50, 100$. Recall that $S = 1$ is the standard ETD1 integtrator.

For sufficiently low Δt we have the predicted results, with the error being first order with respect to Δt, and decreasing as S increases. For Δt too large, this is not the case. Here the Krylov subspace dimension m is most likely the limiting factor as Assumption 3.1 becomes invalid. In Fig. B.6 b and d we show the efficiency by plotting the estimated error against cputime. For Δt low enough that the substepping schemes are effective, the scheme with 10 substeps is the most efficient.

We can see the existence of an optimal S for efficiency, as predicted, in Fig. B.6 d, where the scheme using $S = 50$ is more efficient than the scheme using $S = 100$. Any increase in accuracy by increasing S from 50 to 100 is extremely small (indeed, it is unnoticeable in Fig. B.6 c, and not enough to offset the increase in cputime. In fact, Fig. B.6 d shows that for this experiment the scheme using $S = 10$ is more efficient than both the $S = 50$ and $S = 100$ schemes. Fig. B.6 c shows that the $S = 10$ scheme is also slightly more accurate than both. This is likely because at $S = 10$ the improvement in accuracy is already close to the limiting value, and greatly increasing S to 50 or 100 only accumulates rounding errors without further benefit. Fig. B.6 a shows that the improvement from $S = 1$ to $S = 10$ is quite significant on its own.

In Fig. B.7 we compare the 2-step corrector scheme against the two other second order exponential integrators, ETD2 and ROS2. Fig. B.7 a shows estimated error against Δt, and we see that, like Fig. B.6 a, the Krylov recycling scheme does not function as intended above a certain Δt threshold; again this is due to the timestep being too large with respect to m. The standard exponential integrators do not have this problem, as their timesteps are driven by phipm.m, which takes extra (non-recycled, linear) substeps to achieve a desired error. Below the Δt threshold, the 2-step corrector scheme functions exactly as intended, exhibiting second order convergence and high accuracy. In Fig. B.7 b, we can observe that the 2-step corrector scheme is more efficient than the other two schemes for lower Δt, and of comparable efficiency for larger Δt.

It is interesting to compare Fig. B.6 a and Fig. B.7 a and note that the threshold Δt for the corrector scheme seems to be lower than for the substepping schemes.

In Fig. B.6 b we can again see that for a fixed cputime, increasing S from 1 to 10 decreases error by roughly one order of magnitude; however Fig. B.6 d shows no improvement in increasing S from 10 to 50. Comparing Fig. B.6 b and Fig. B.7 b shows that the second order corrector scheme can be almost three orders of magnitude more accurate for a fixed cputime than the first order recycling scheme with $S = 10$.

In Fig. B.3 we show the alternative measure of the efficiency for the S-step recycling scheme where the logarithm of error per unit time is plotted against the logarithm of S. We can see such a minimum at $S = 2$ for the $\Delta t = 0.024$ curve, indicating an optimal value of S there.

5.3. Large 2D example with random fields

In this example the 2D grid models a domain with physical dimensions $100 \times 100 \times 10$; the grid is split into a $1000 \times 1000 \times 1$ cells. The model equation is the same as the previous example (54). The diffusivity is kept constant at $D = 0.01$, while a random velocity V field is used. For this we generated a random permativity field K, which was then used to generate a corresponding pressure field and then a velocity field in a standard way, using Darcy's Law, see [25,31]. The pressure p field was determined by the permativity field and the Dirichlet boundary conditions $p = 1$ where $y = 0$ and $p = 0$ where $y = 100$. The initial conditions for v were zero everywhere, and the boundary conditions were the same as for p, $v = 1$ where $y = 0$ and $v = 0$ where $y = 100$. The final time was $T = 500$.

Due to the large size of system (10^6 unknowns) we only examine the recycling scheme and for a system of this size, it was necessary to increase m to 100 to prevent the Krylov error from being dominant. The results are shown in Figure B.10. We see that, for Δt sufficiently low, increasing S decreases the error and increases the efficiency of the scheme. The improvement in efficiency between $S = 5$ and $S = 10$ is marginal; the optimal S for this example would not be much greater than 10. This is also indicated by the alternative efficiency measure in Fig. B.11.

5.4. A 3D example with fracture

Here we consider a three dimensional example with a randomly generated fracture, as in the two dimensional example in Section 5.2. In this example the diffusivity is set to $D = 1$ except for in the fracture where it is set to $D = 10$. We set a global velocity field of $V = (0.1, 0, 0.1)^T$, and no-flow boundary conditions on every boundary face. The final time is $T = 10$, and the the domain is a $10^3 m$ cube discretised into a 15^3 cell grid. The initial condition was $c(\mathbf{x}) = 0$ everywhere except at $\mathbf{x} = (0, 0, 0)^T$ where $c(\mathbf{x}) = 1$. The reaction term is the same Langmuir type reaction (54) as in Section 5.2 and Section 5.3. Our schemes used $m = 100$ for the Krylov subspace size.

We show the final state of the system in Fig. B.12. Plots (a) and (b) show the surface of the domain, while the cutaways (c) and (d) show just the fracture cells. Plots (a) and (c) are the comparison solve, generated with ETD and $\Delta t = 10^{-4}$, and plots (b) and d) are the solve produced by the $S = 10$ substepping scheme with $\Delta t = 1$.

The standard error and efficiency plots are given in Fig. B.13 and Fig. B.14. We note that all the errors for this system are quite small. In addition, the difference in error between schemes with different values of S is quite small, and for this system the improvement of in accuracy from increasing S is less significant than the other examples, though can still be observed. Due to the expected diminishing returns with increasing S, we see that the $S = 10$ solve is less efficient that the $S = 5$, with the optimal S clearly somewhere between the two values.

It may be contested that there is little practical use for the substepping in this situation. Indeed, we have found for some examples with certain values of m that it is possible for increasing S to have no benefit at all, or to slightly increase error

(likely due to accumulated rounding errors due to the increasing number of operations). This illustrates how a more robust, adaptive algorithm based on the substepping technique and the error results presented above would be required in practice, to identify and use optimal values of S and m in order to get the most efficiency out of each Krylov subspace and timestep. Working towards such an algorithm would be an interesting avenue of future research.

6. Conclusions

We have extended the notion of recycling a Krylov subspace for increased accuracy in the sense of [26]. We have applied this new method to the first order ETD1 scheme and examined the effect of taking an arbitrary number of substeps (the parameter S). The local error has been expressed in terms of S, and the expression shows that the local error will decrease with S down to a finite limit. The discussion in Appendix B examines construction for EEM. Results suggest that there maybe an optimal S for a maximal efficiency increase and some preliminary analysis in this direction may be found in [25]. Convergence and existence of an optimal $S > 1$ has been demonstrated with numerical experiments. Additional information from the substeps was used to form a corrector and a second order scheme. This was shown to be comparable to, or slightly better than, ETD2 and ROS2 in our tests.

The schemes currently rely on Assumption 3.1, essentially requiring that Δt be sufficiently small and m be sufficiently large, to be effective. Numerical experiments have shown how having Δt too large can cause the schemes to become inaccurate as the error of the initial Krylov approximation becomes significant. It is already well established how the Krylov approximation error can be controlled by adapting m and the use of non-recycling substeps. Applying these techniques to the schemes presented here in future work would allow them to be effective over wider Δt ranges.

Acknowledgements

The authors are grateful to Prof. S. Geiger for his input into the flow simulations. The work of Dr. D. Stone was funded by the SFC/EPSRC(EP/G036136/1) as part of NAIS.

Appendix A. Matlab implementation

We show in Algorithm 1 the simple Matlab code used to implement the first order recycling method. Note that we call the function phipade as a dependency, this function computes φ-functions using a standard pade method and is part of the expint package (https://www.math.ntnu.no/num/expint/) [22].

Appendix B. substepping with the scheme EEM

The method of recycled Krylov subspace recycling that we have examained was introduced in [26], where the second order exponential integrator EEM with one recycled step (i.e., $S = 2$) was investigated. Continuing from this, we now show how to apply our analysis to EEM for arbitrary S.

Applied to the system of ODEs

$$\frac{du}{dt} = g(u),$$

where $g(u)$ may not be semilinear, the scheme EEM is given by

$$u_{n+1} = u_n + \Delta t \varphi_1(\Delta t J_n) g_n, \tag{B.1}$$

where J denotes the Jacobian of g and $J_n = J(u_n)$. The Jacobian J_n is kept fixed for the entire step Δt, including recycling substeps (again, see [26] for more details of the scheme setup). Therefore an S step recycling scheme can be defined on EEM in exactly the same way as the recycling scheme for ETD1. Note that the Krylov subspace will be generated for J and g in the EEM case, i.e. $\mathcal{K} = \mathcal{K}(J_n, g_n)$.

With $\tilde{p}_\tau \equiv \tau V_m \varphi_1(\tau H_m) V_m^T$ approximating $\tau\varphi_1(\tau J)$ Applying the Krylov subspace recycling scheme to EEM we have

$$u_{n+\frac{j}{S}} = u_n + \tilde{p}_{j\delta t} g_n + R^S_{n+\frac{j}{S}}.$$

Following the same steps as in Lemma 3.3 we obtain the following result.

Corollary B.1. *The remainder $R^S_{n+\frac{j}{S}}$ satisfies the recursion relation*

$$R^S_{n+\frac{j+1}{S}} = \tilde{p}_{\delta t}(F_{n+\frac{j}{S}} - F_n) + (I + \tilde{p}_{\delta t} L) R^S_{n+\frac{j}{S}}, \tag{B.2}$$

where $F_{n+\frac{j+1}{S}} = g_{n+\frac{j+1}{S}} - J_n u_{n+\frac{j+1}{S}}$.

Algorithm 1. The Matlab implementation of the recycling scheme.

```
function [U]=kry_rec_x_step(dt,L,U0,T,krydim,f, x)

% to approximate the ODE system
% du/dt = L u + f(t)  (1)

% Input:
% dt: fixed timestep length
% L: matrix from eq (1)
% U0: initial condition
% T: final time
% krydim: i.e. m, the dimension of the Krylov subspace to use
% f: a function, the f from equation (1)
% x: the number of recycled substeps to use. Setting x = 1 will
% result in a method equivalent to ETD1.

N=length(U0);
padeg = 20;
e1  = zeros(krydim,1);
e1(1) = 1;

ddt = dt/x;
xm = x-1;

maxits = ceil(T/dt);

U = U0;

for k = 1:maxits

    F = f(U);
    RHS = L*U + F;
    [H,V,beta] = Arnoldi(RHS,N,L,krydim);
    VT = V.';
    phimat = phipade(ddt*H, 1,padeg); % dependency
    ddtVphimat = ddt*V*phimat;
    U = U + beta*ddtVphimat*e1; % first step is not recycled

    for j = 1:xm % recycled steps
        F = f(U);
        RHS = L*U + F;

        step1 = VT*RHS;
        U = U + ddtVphimat*step1;
    end

end
```

To examine the remainder term in more detail let $\hat{J}_i$ be the Hessian matrix

$$\hat{J}_i = \begin{pmatrix} (\hat{g}_i)_{x_1x_1} & (\hat{g}_i)_{x_1x_2} & \dots \\ (\hat{g}_i)_{x_2x_1} & (\hat{g}_i)_{x_2x_2} & \dots \\ \dots & \dots & \dots \end{pmatrix},$$

where $\hat{g}_i$ is the ith entry of the vector g. Let the tensor $\hat{\mathbf{J}}$ be a vector with the matrix $\hat{J}_i$ in its ith entry. We can now Taylor expand the remainder $R^S_{n+\frac{j}{S}}$ from (B.2) to find the local error of the EEM scheme with recycled substeps.

Lemma B.2. *For the EEM recycling scheme, the leading term of* $R^S_{n+\frac{j}{S}}$ *satisfies*

$$R^S_{n+\frac{j}{S}} = \alpha(j)\delta t^3 V_m V_m^T g_n^T \hat{\mathbf{J}} g_n + O(\delta t^4)$$

where $\alpha(j) = (2j^3 - 3j^2 + j)/24$.

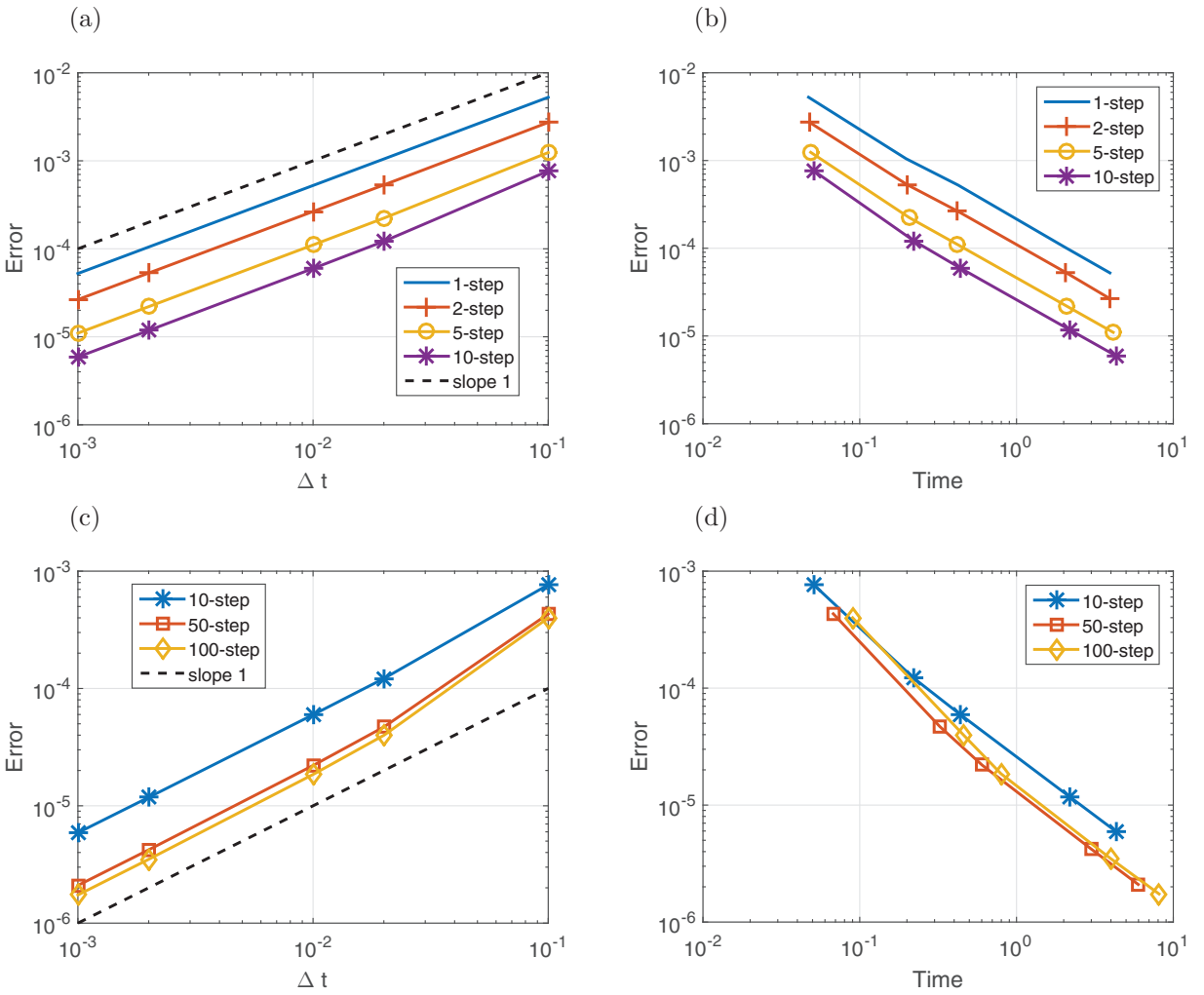

Fig. B1. Results for the substepping schemes applied to the Allen–Cahn type system. (a) and (c) display estimated error against timestep Δt. (b) and (d) display estimated error against cputime, showing efficiency.

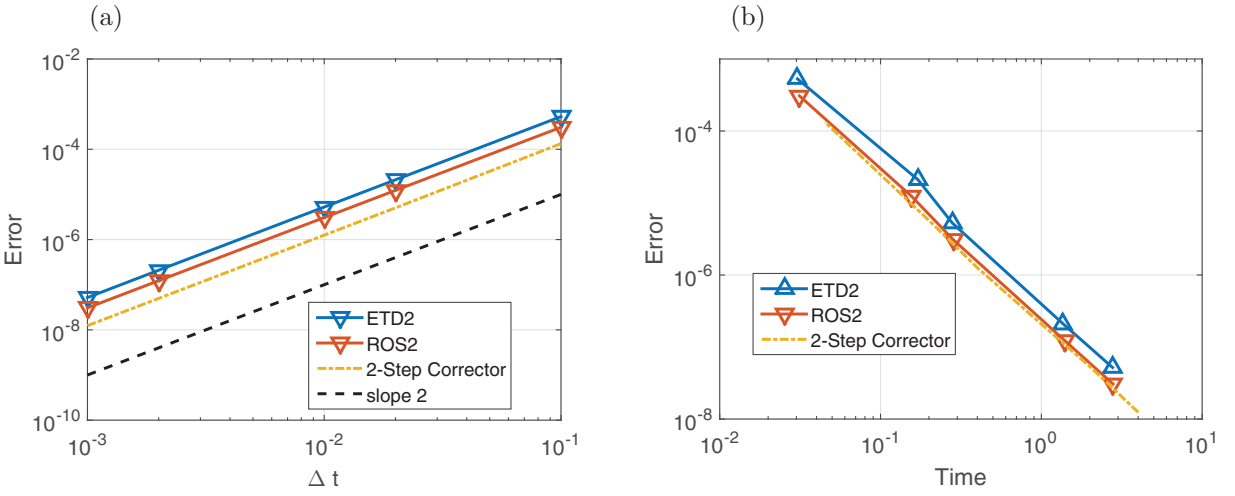

Fig. B2. AC system, Comparing the second order recycling-corrector scheme with ETD2 and ROS2. (a) Estimated error against timestep Δt. (b) Estimated error against cputime.

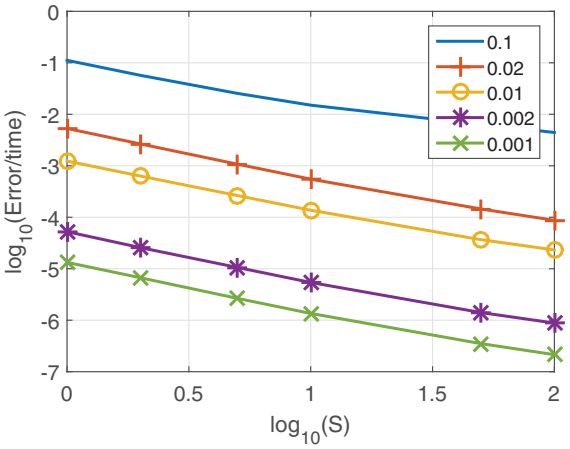

Fig. B3. An alternative measure of the efficiency for the S-step recycling scheme for the experiment in Section 5.1. The logarithm of error per unit time is plotted against the logarithm of S. Each curve is a different fixed timestep Δt value. Minima in the curves would indicate an optimal S for efficiency. In this case, for every timestep the optimal S would be greater than 100.

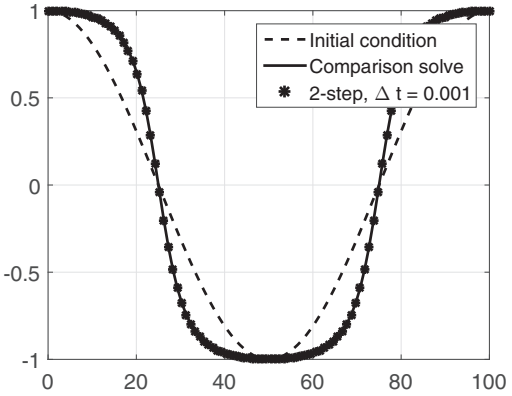

Fig. B4. Showing the comparison solve and a result using the recycling scheme for the AC system.

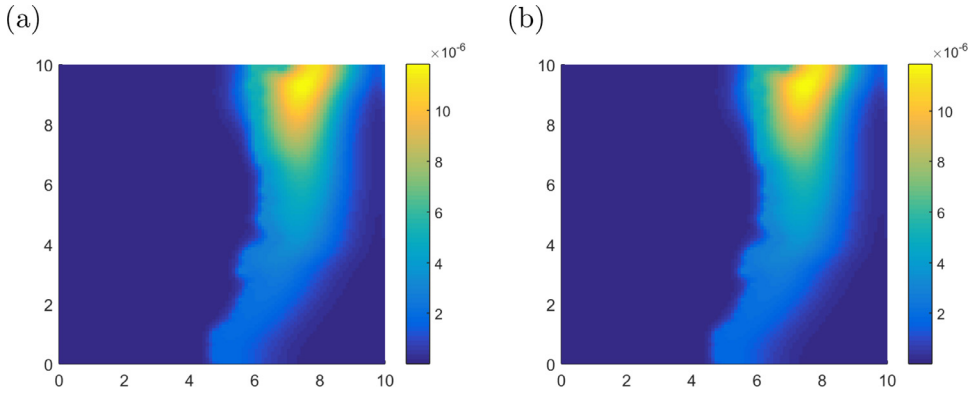

Fig. B5. The final state of the fracture system with Langmuir type reaction. (a) Result produced by the 2-step scheme with $\Delta t = 2.4 \times 10^{-4}$. (b) Result produced by ETD2 with $\Delta t = 2.4 \times 10^{-5}$.

Proof. By induction. The base case is true for $j = 1$ with $\alpha(1) = 0$ since there is no recycling at that step. Assume true for some j. Consider $g_{n+\frac{j}{S}}$,

$$g_{n+\frac{j}{S}} = g(u_{n+\frac{j}{S}}) = g(u(t) + j\delta t g(t) + \frac{1}{2}(j\delta t)^2 Jg + O(\delta t^3)),$$

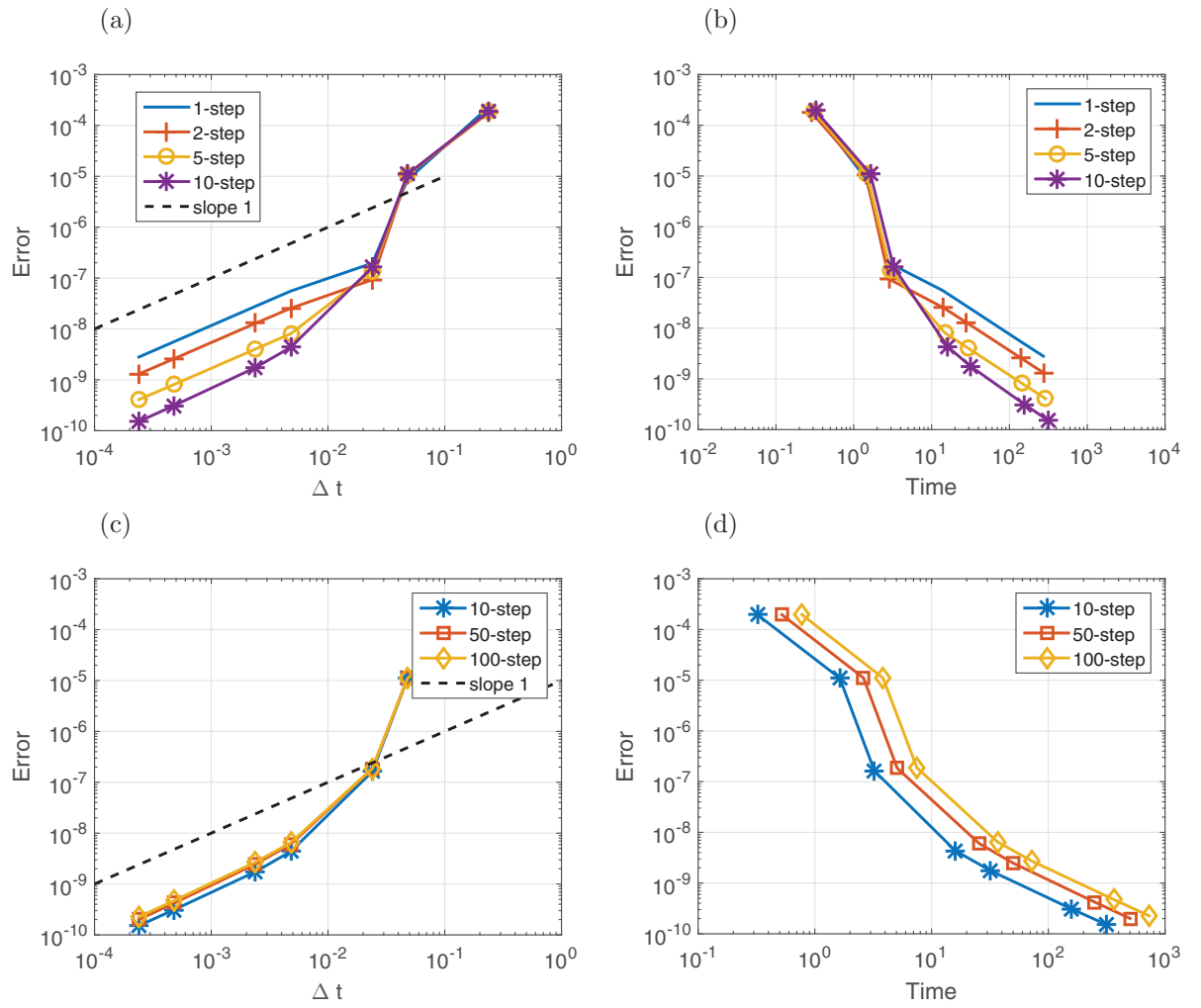

Fig. B6. Results for the substepping schemes applied to the Langmuir type reaction system. (a) and (c) display estimated error against timestep Δt. (b) and (d) display estimated error against cputime, showing efficiency. In (c), points for the 50 step scheme are marked with circles, and points with the 100 step scheme are marked with triangles, to help distinguish the (very similar) results for the two schemes. This is also done in plot (d) for consistency.

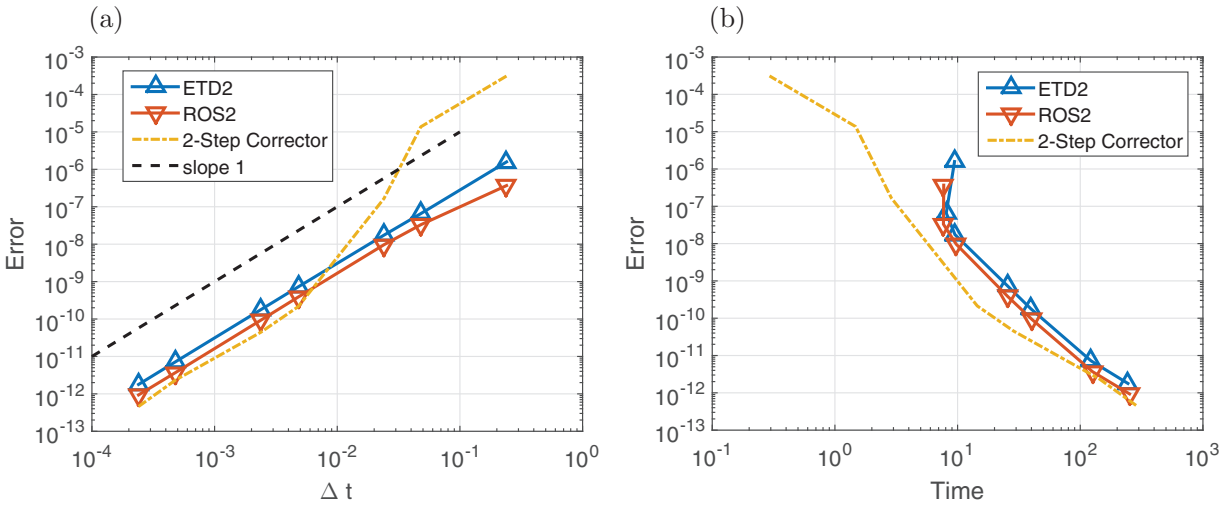

Fig. B7. Langmuir type reaction system, Comparing the second order recycling-corrector scheme with ETD2 and ROS2. (a) Estimated error against timestep Δt. (b) Estimated error against cputime.

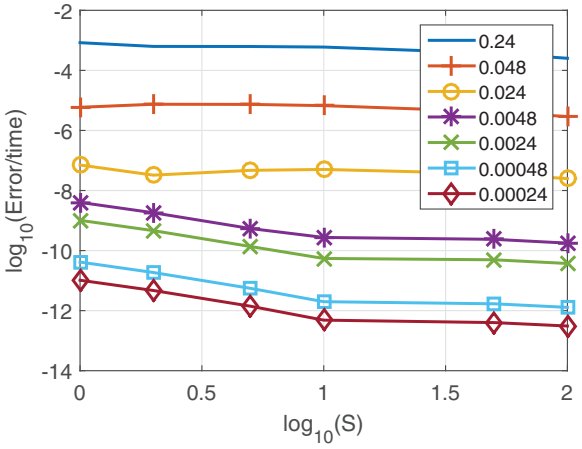

Fig. B8. An alternative measure of the efficiency for the S-step recycling scheme for the experiment in Section 5.2. The logarithm of error per unit time is plotted against the logarithm of S. Each curve is a different fixed timestep Δt value. Minima in the curves would indicate an optimal S for efficiency. We can see such a minimum at $S = 2$ for the $\Delta t = 0.024$ curve.

to second order this is

$$g_{n+\frac{j}{S}} = g_n + J\left(j\delta t g_n + \frac{1}{2}(j\delta t)^2 J g_n\right) + \frac{(j\delta t)^2}{2} g_n^T \hat{\mathbf{J}} g_n + O(\delta t^3),$$

where we have made use of the local error assumption $u(t_n) = u_n$. Then,

$$F_{n+\frac{j}{S}} - F_n = g_{n+\frac{j}{S}} - g_n - J u_{n+\frac{j}{S}} + J u_n$$

and since $u_{n+\frac{j}{S}} = u_n j\delta t g_n + \frac{(j\delta t)^2}{2} J g_n + O(\delta t^3)$ up to second order and the induction hypothesis we have

$$F_{n+\frac{j}{S}} - F_n = \frac{(j\delta t)^2}{2} g_n^T \hat{\mathbf{J}} g_n + O(\delta t^3).$$

Now consider

$$\tilde{p}_{\delta t}\left(F_{n+\frac{j}{S}} - F_n\right) = \frac{j^2(\delta t)^3}{2} V_m V_m^T g_n^T \hat{\mathbf{J}} g_n + O(\delta t^4)$$

The induction relation for $R^S_{n+\frac{j+1}{S}}$ then gives us

$$R^S_{n+\frac{j+1}{S}} = \frac{j^2(\delta t)^3}{2} V_m V_m^T g_n^T \hat{\mathbf{J}} g_n + (I + \tilde{p}_{\delta t} J) R^S_{n+\frac{j}{S}} + O(\delta t^4)$$

which to leading order this is

$$R^S_{n+\frac{j+1}{S}} = \left(\frac{j^2}{2} + \alpha(j)\right) \delta t^3 V_m V_m^T g_n^T \hat{\mathbf{J}} g_n + O(\delta t^4).$$

So $\alpha(j+1) = \frac{j^2}{2} + \alpha(j)$, $\alpha(1) = 0$. which is satisfied by the given $\alpha(j)$. □

We now combine the leading term of the remainder R and the known local error of EEM

$$\frac{1}{6}\Delta t^3 g^T \hat{\mathbf{J}} g.$$

(see, for example, [25]) to find the local error of the new recycling scheme.

Corollary B.3. *The leading term of the local error of the S step recycling scheme for EEM at the end of a timestep is*

$$\frac{\Delta t^3}{6}\left(I - \left(\frac{2S^2 - 3S + 1}{2S^2}\right) V_m V_m^T\right) g^T \hat{\mathbf{J}} g.$$

From this, we can predict similar properties to the ETD1 recycling scheme. This extends the work of [26], where the recycling substepping EEM scheme was used for a single substep.

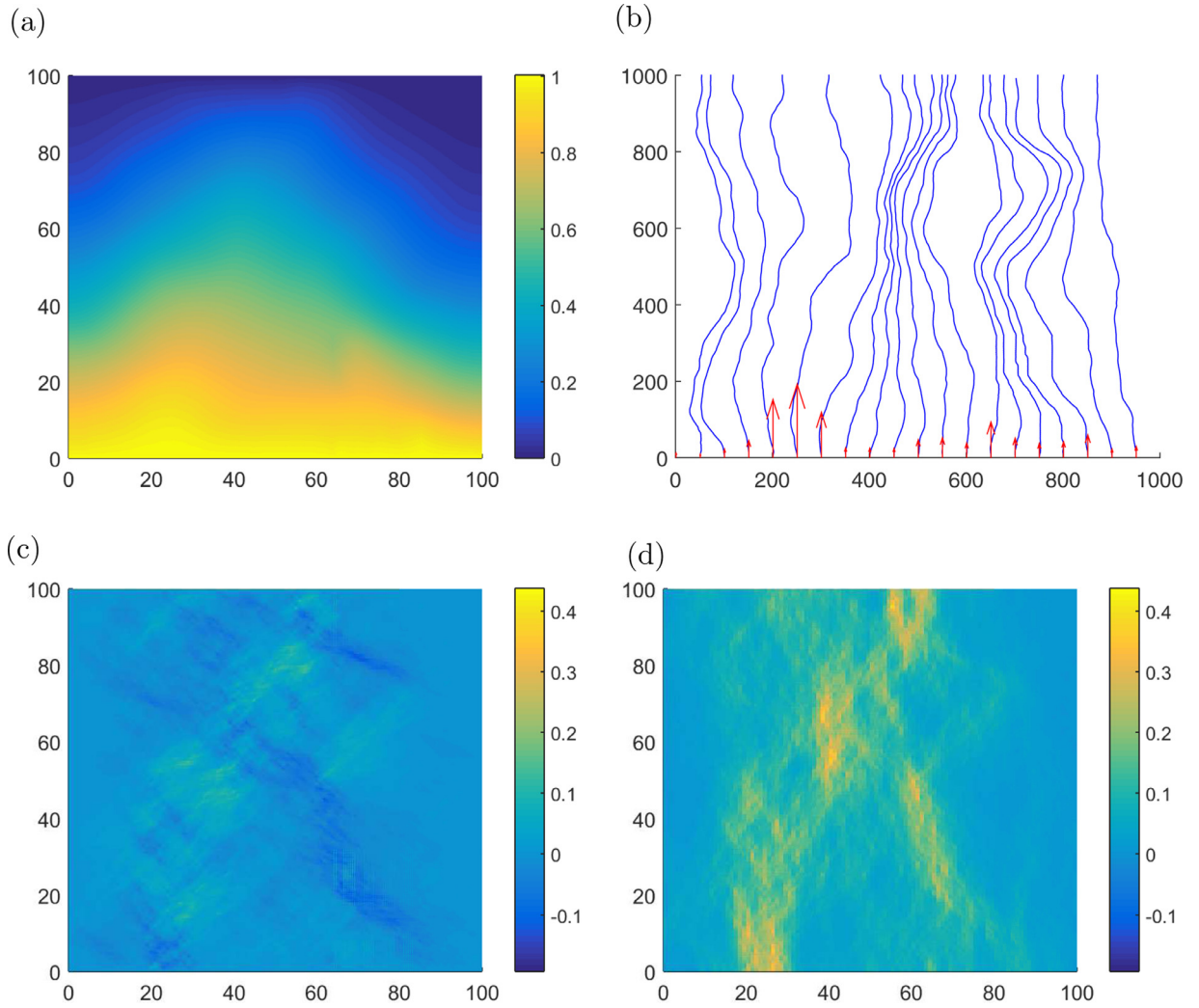

Fig. B9. Result for the example in Section 5.3, in which solute enters through the lower boundary and flows according to a random velocity field. Produced by the 10-step recycling scheme with $\Delta t = 0.2441$, i.e. 2048 steps. (a) Shows the system at the final time $T = 500$; the axes indicates the physical dimensions (i.e., the domain is 100 $\times$ 100 metres). (b) shows streamlines for the velocity field; the axes indicate the cells in the finite volume grid (i.e., the grid has 1000 cells along each side). (c) and (d) show the x and y components of the velocity field, respectively (the axes here also show the finite volume grid dimensions).

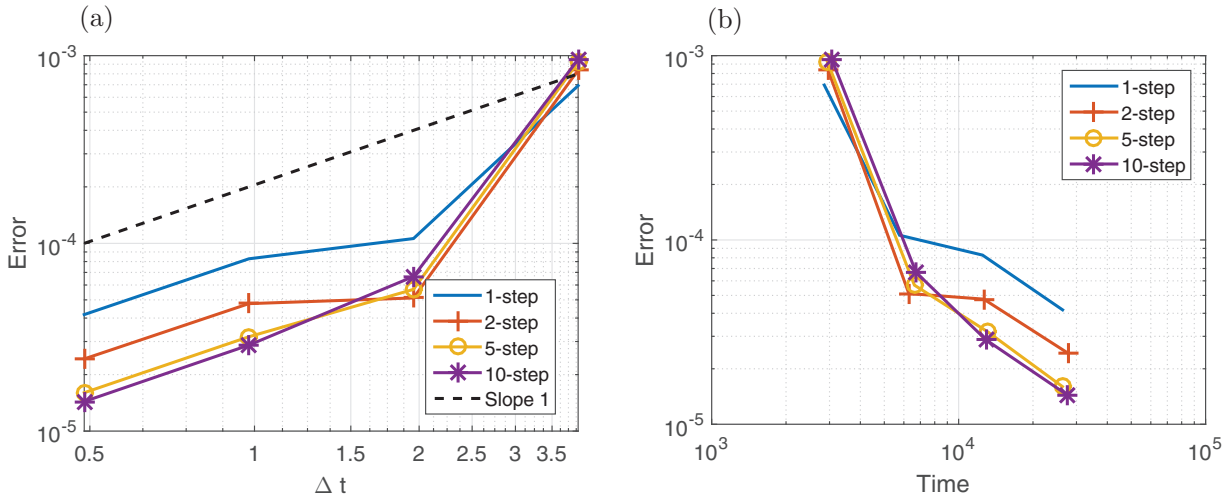

Fig. B10. Results for the substepping scheme applied to the large Langmuir type reaction system in Section 5.3. (a) Estimated errors against timestep Δt and (b) displays estimated error against cputime, showing efficiency. Note that for the largest timestep the error is dominated by the Krylov error as m is too small for the given Δt (c.f. Assumption 3.1).

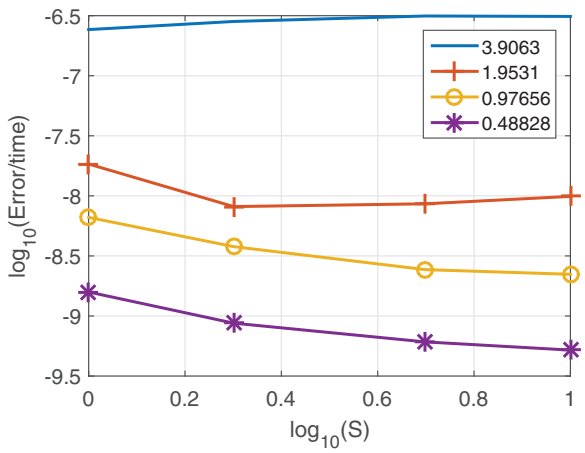

Fig. B11. An alternative measure of the efficiency for the S-step recycling scheme for the experiment in Section 5.3. The logarithm of error per unit time is plotted against the logarithm of S. Each curve is a different fixed timestep Δt value. Minima in the curves would indicate an optimal S for efficiency.

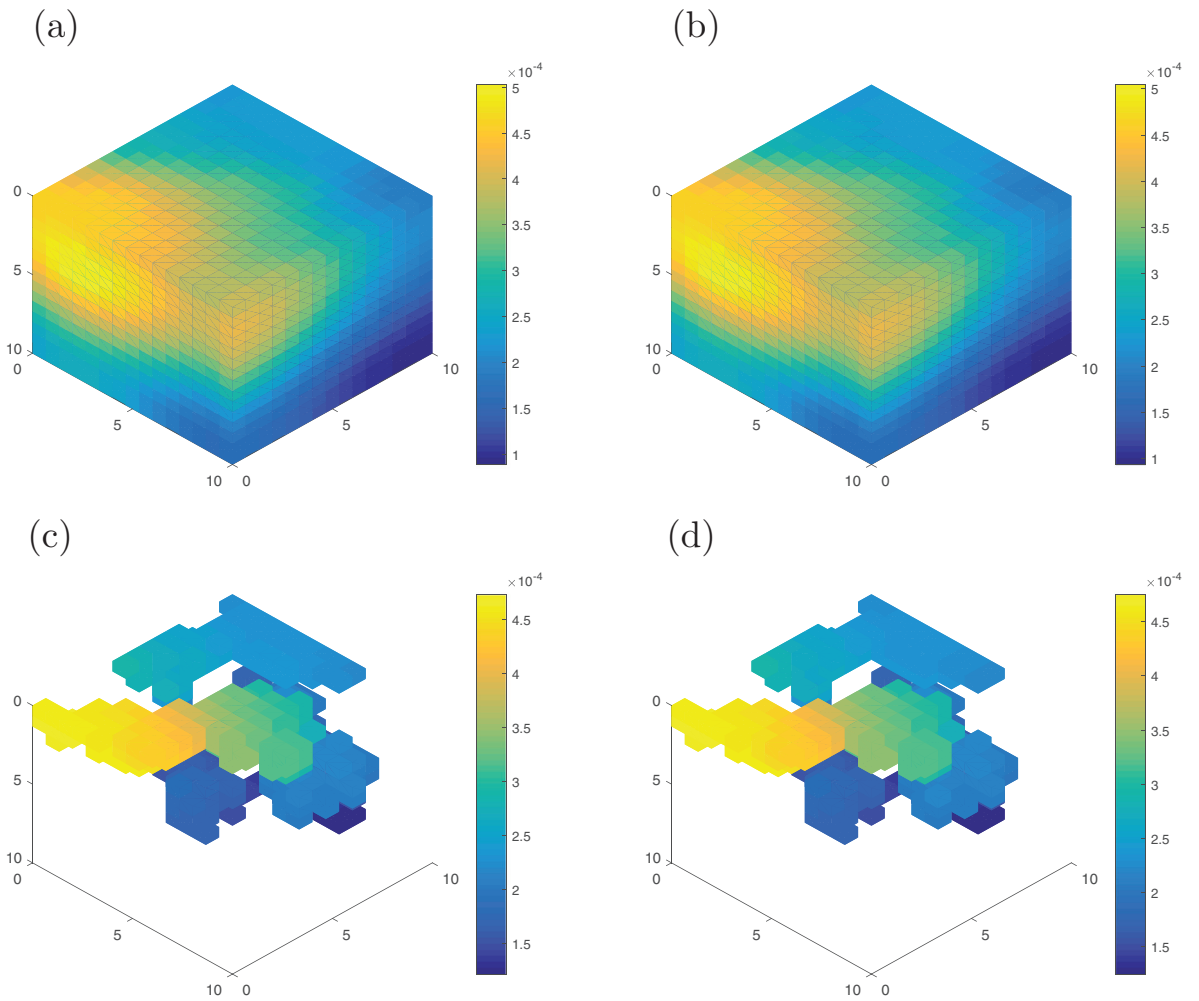

Fig. B12. Result for the example in Section 5.4. Top row (a and b) show the surface of the cube domain, bottom row (c and d) show only the cells assigned to be part of the 'fracture'. Left column plots (a and c) are produced using the comparison solve (ETD2), and the right column plots are produced using the $S = 10$ recycling solver with $\Delta t = 1$.

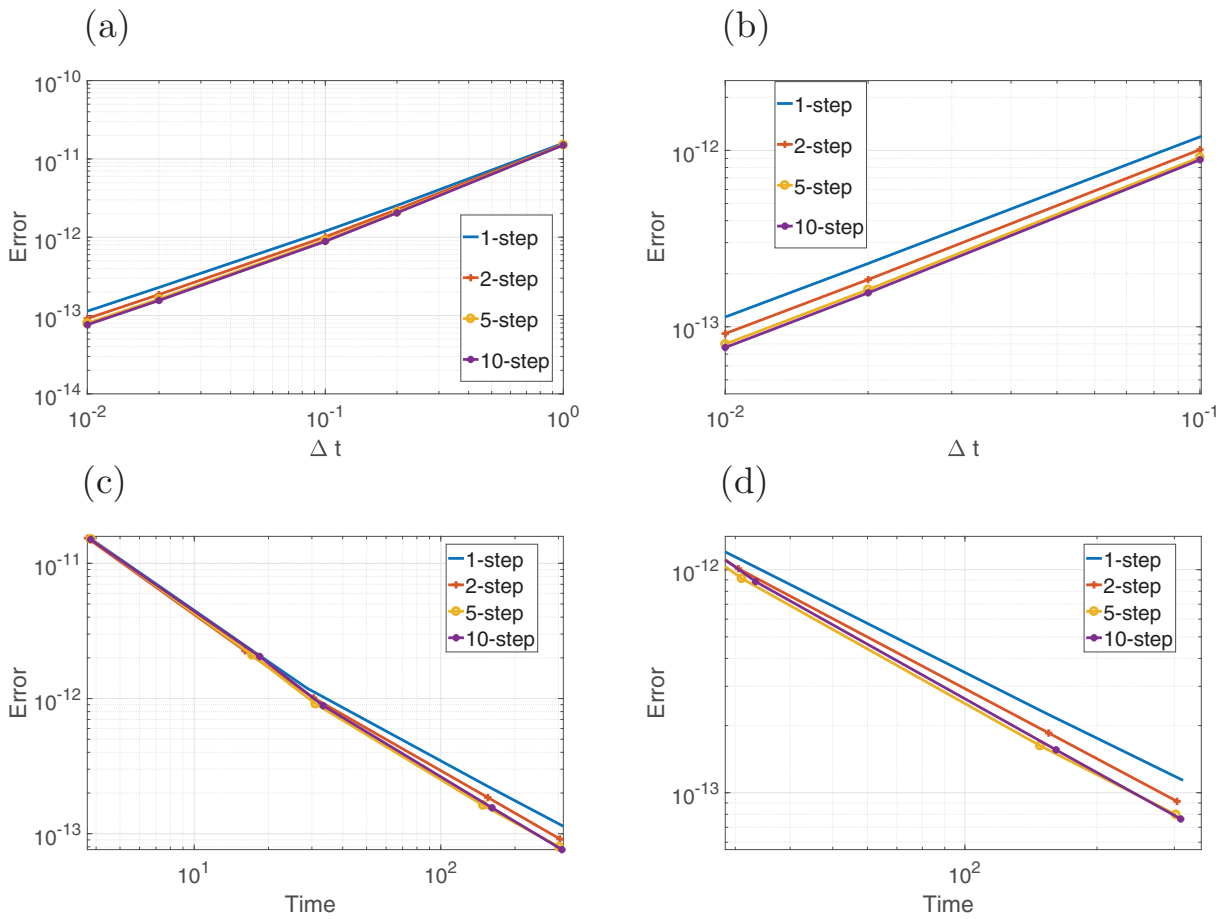

Fig. B13. Result for the example in Section 5.4. (a) Timestep against estimated error. (b) Zoomed in portion of (a). (c) Time against estimated error. (d) Zoomed in portion of (c).

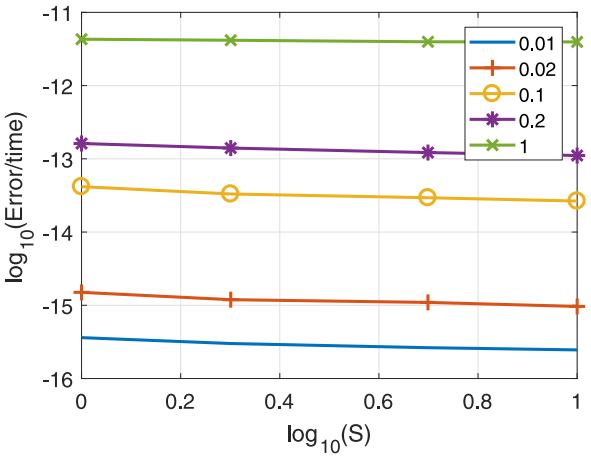

Fig. B14. An alternative measure of the efficiency for the S-step recycling scheme for the experiment in Section 5.4. The logarithm of error per unit time is plotted against the logarithm of S. Each curve is a different fixed timestep Δt value. Minima in the curves would indicate an optimal S for efficiency.

References

[1] S.M. Cox, P.C. Matthews, Exponential time differencing for stiff systems, J. Comput. Phys. 176 (2002) 430–455.
[2] M. Caliari, M. Vianello, L. Bergamaschi, The LEM exponential integrator for advection-diffusion-reaction equations, J. Comput. Appl. Math. 210 (1–2) (2007) 56–63, doi:10.1016/j.cam.2006.10.055.
[3] M. Hochbruck, A. Ostermann, J. Schweitzer, Exponential Rosenbrock-type methods, SIAM J. Numer. Anal. 47 (1) (2009) 786–803.
[4] M. Caliari, A. Ostermann, Implementation of exponential Rosenbrock-type integrators, Appl. Numer. Math. 59 (3–4) (2009) 568–581, doi:10.1016/j.apnum.2008.03.021.
[5] M. Hochbruck, A. Osterman, Exponential integrators, Acta Numerica (2010) 209–286.
[6] B. Minchev, W. Wright, A review of exponential integrators for first order semilinear problems, Institution: Norwegian University of Science and Technology, 2005 https://www.math.ntnu.no/preprint/numerics/2005/N2-2005.pdf.
[7] A.K. Kassam, L.N. Trefethen, Fourth-order time-stepping for stiff PDEs, SIAM J. Sci. Comput. 26 (4) (2005) 1214–1233 (electronic), doi:10.1137/S1064827502410633.
[8] E. Carr, I. Turner, Two-scale computational modelling of water flow in unsaturated soils containing irregular-shaped inclusions, Int. J. Numer. Methods Eng. 98 (3) (2014) 157–173.
[9] E.J. Carr, I.W. Turner, P. Perré, A variable-stepsize Jacobian-free exponential integrator for simulating transport in heterogeneous porous media: application to wood drying, J. Comput. Phys. 233 (2013) 66–82.
[10] A. Tambue, I. Berre, J.M. Nordbotten, Efficient simulation of geothermal processes in heterogeneous porous media based on the exponential Rosenbrock–Euler and Rosenbrock-type methods, Adv. Water Resour. 53 (2013) 250–262.
[11] A. Tambue, G.J. Lord, S. Geiger, An exponential integrator for advection-dominated reactive transport in heterogeneous porous media, J. Comput. Phys. 229 (10) (2010) 3957–3969.
[12] C. Moler, C. Van Loan, Nineteen dubious ways to compute the exponential of a matrix, SIAM Rev. 20 (4) (1978) 801–836, doi:10.1137/1020098.
[13] A.H. Al-Mohy, N.J. Higham, Computing the action of the matrix exponential, with an application to exponential integrators, SIAM J. Sci. Comput. (2011).
[14] I. Moret, P. Novati, An interpolatory approximation of the matrix exponential based on Faber polynomials, J. Comput. Appl. Math. 131 (1–2) (2001) 361–380, doi:10.1016/S0377-0427(00)00261-2.
[15] J. Baglama, D. Calvetti, L. Reichel, Fast Leja points, Electron. Trans Numer. Anal. 7 (1998) 124–140. Large scale eigenvalue problems (Argonne, IL, 1997).
[16] L. Bergamaschi, M. Caliari, M. Vianello, The ReLPM exponential integrator for FE discretizations of advection-diffusion equations, in: Computational Science—ICCS. Part IV, in: Lecture Notes in Computer Science, 3039, Springer, Berlin, 2004, pp. 434–442, doi:10.1007/978-3-540-25944-2_57.
[17] A. Martínez, L. Bergamaschi, M. Caliari, M. Vianello, A massively parallel exponential integrator for advection-diffusion models, J. Comput. Appl. Math. 231 (1) (2009) 82–91, doi:10.1016/j.cam.2009.01.024.
[18] L. Bergamaschi, M. Caliari, A. Martinez, M. Vianello, Comparing Leja and Krylov approximations of large scale matrix exponentials, in: Computational Science–ICCS, Springer, 2006, pp. 685–692.
[19] M. Caliari, M. Vianello, L. Bergamaschi, Interpolating discrete advection-diffusion propagators at Leja sequences, J. Comput. Appl. Math 172 (1) (2004) 79–99, doi:10.1016/j.cam.2003.11.015.
[20] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 29 (1) (1992) 209–228, doi:10.1137/0729014.
[21] M. Hochbruck, C. Lubich, On Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. 34 (5) (1997) 1911–1925, doi:10.1137/S0036142995280572.
[22] R.B. Sidje, Expokit: a software package for computing matrix exponentials, ACM Trans. Math. Softw. 24 (1) (1998) 130–156.
[23] J. Niesen, W.M. Wright, Algorithm 919: a Krylov subspace algorithm for evaluating the ϕ-functions appearing in exponential integrators, ACM Trans. Math Softw. 38 (3) (2012) 22.
[24] M. Tokman, J. Loffeld, Efficient design of exponential-Krylov integrators for large scale computing, Procedia Comput. Sci. 1 (1) (2010) 229–237.
[25] D. Stone, Asynchronous and exponential based numerical schemes for porous media flow, Heriot-Watt, 2015 Ph.D. thesis.
[26] E. Carr, T. Moroney, I. Turner, Efficient simulation of unsaturated flow using exponential time integration, Appl. Math. Comput. 217 (2011) 6587–6596.
[27] E. Isaacson, H.B. Keller, Analysis of Numerical Methods, Courier Corporation, 2012.
[28] J.C. Butcher, Numerical methods for ordinary differential equations, 2nd ed., John Wiley & Sons, Ltd., Chichester, 2008, doi:10.1002/9780470753767.
[29] M. Hochbruck, A. Ostermann, Explicit exponential Runge-Kutta methods for semilinear parabolic problems, SIAM J. Numer. Anal. 43 (3) (2006) 1069–1090 (electronic), doi:10.1137/040611434.
[30] Y. Saad, Iterative Methods for Sparse Linear Systems, ITP, 1996.
[31] K.A. Lie, S. Krogstad, I.S. Ligaarden, J.R Natvig, H.M. Nilsen, B. Skaflestad, Open-source Matlab implementation of consistent discretisations on complex grids, Comput. Geosci. 16 (2) (2012) 297–322.